



DEVELOPING A FREE DLP SYSTEM FOR EMAIL DATA LEAKAGE

KORAY YILMAZ

JUNE 2017

DEVELOPING A FREE DLP SYSTEM FOR EMAIL DATA LEAKAGE

A THESIS SUBMITTED TO
THE GRADUATE SCHOOL OF NATURAL AND APPLIED
SCIENCES OF
ÇANKAYA UNIVERSITY

BY
KORAY YILMAZ

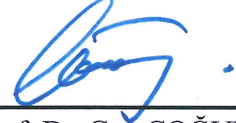
IN PARTIAL FULFILMENT OF THE REQUIREMENTS FOR THE
DEGREE OF
MASTER OF SCIENCE
IN
THE DEPARTMENT OF
COMPUTER ENGINEERING

JUNE 2017

Title of the Thesis: **Developing a Free DLP System for Email Data Leakage.**

Submitted by **Koray YILMAZ**

Approval of the Graduate School of Natural and Applied Sciences, Çankaya University.



Prof. Dr. Can ÇOĞUN

Director

I certify that this thesis satisfies all the requirements as a thesis for the degree of Master of Science.



Prof. Dr. Erdoğan DOĞDU
Head of Department

This is to certify that we have read this thesis and that in our opinion it is fully adequate, in scope and quality, as a thesis for the degree of Master of Science.

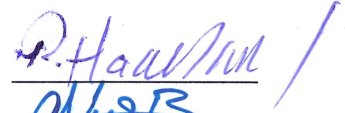


Asst. Prof. Dr. A. Nurdan SARAN
Supervisor

Examination Date: 05.05.2017

Examining Committee Members

Asst. Prof. Dr. Reza Z. HASSANPOUR (Çankaya Univ.)



Asst. Prof. Dr. A. Nurdan SARAN (Çankaya Univ.)



Asst. Prof. Dr. Gönenç ERCAN (Hacettepe Univ.)



STATEMENT OF NON-PLAGIARISM PAGE

I hereby declare that all information in this document has been obtained and presented in accordance with academic rules and ethical conduct. I also declare that, as required by these rules and conduct, I have fully cited and referenced all material and results that are not original to this work.

Name, Last Name : Koray YILMAZ

Signature : 

Date : 05.05.2017

ABSTRACT

DEVELOPING A FREE DLP SYSTEM FOR EMAIL DATA LEAKAGE

YILMAZ, Koray

M.Sc., Department of Computer Engineering

Supervisor: Asst. Prof. Dr. A. Nurdan SARAN

June 2017, 71 pages

Data leakage (or data loss) is a term used in the information security field to describe unwanted disclosures of information. According to DataLossDB site data breaches (leakages) in 2015 are increased dramatically. According to Risk Based Security 2016 Data Breach Report analysis, while the percentage of total breaches impact 89.5% electronic data, it is 99.6% in 2016.

In this thesis, a lite e-mail data leakage prevention (DLP) solution using open source projects for e-mail data leakage has been proposed. A dirty filter script running on top of the EmailRelay server is developed and the system is tested under different loads. Different parsing architectures are tried and searched for the most efficient one. Also the proposed solution is compared with another known solution. For this purpose the community version of MyDLP server is evaluated.

The outgoing e-mail with binary attachment content is analysed with Apache TIKA framework. The e-mail content which is converted to textual data, is searched for data leakage patterns using regular expressions. In most of the studies the scanned image documents are not analysed for leakage. In this study open source tools are used to convert the scanned image document to textual data; because most of the documents in the Internet are scanned confidential documents. Finally, the leaked e-mail is indexed with Elasticsearch for further contextual analysis.

Keywords: data leakage, e-mail filter, DLP, content analysis

ÖZ

E-POSTA VERİ SIZINTILARI TESPİTİ İÇİN AÇIK KAYNAK VSE SİSTEMİ GELİŞTİRİLMESİ

YILMAZ, Koray

Yüksek Lisans, Bilgisayar Mühendisliği Anabilim Dalı

Tez Yöneticisi : Yrd. Doç. Dr. A. Nurdan SARAN

Haziran 2017, 71 sayfa

Veri sızıntısı (veya veri kaybı), bilgi güvenliği alanında bilginin istenmeden ifşası anlamında kullanılan bir terimdir. DataLossDB sitesine göre 2015'de veri sızıntıları dramatik bir şekilde artmıştır. Risk Based Security 2016 Veri Sızıntı Raporu'na göre 2015'te elektronik veri sızıntıları 89,5% iken, bu değer 2016'da 99,6% olmuştur.

Bu tezde açık kaynak projeleri kullanılarak düşük özellikli bir e-posta veri sızıntı engelleme (VSE) sistemi önerilmiştir. EmailRelay sunucu üzerinde çalışan basit bir filtre betiği geliştirilmiş ve değişik yükler altında sistem test edilmiştir. Değişik ayrıştırma mimarileri denenmiş ve en etkin çalışanı bulunmaya çalışılmıştır. Ayrıca önerilen çözüm başka bir çözümle de karşılaştırılmıştır. Bu kapsamda MyDLP sunucunun topluluk sürümü test edilmiştir.

Dıřarı gnderilen eki olan e-postalar Apache TIKa atısı ile analiz edilmiřtir. Metin veri haline dnřtrlen e-posta ieriđi, dzenli ifadeler kullanılarak veri sızıntı rntleri aranmıřtır. Birok alıřmada taranmıř dokmanlar veri sızıntısı aısından da incelenmemektedir. Bu alıřmada imaj tipi dokmanlardan metin tipi elde etmek iin aık kaynak aralar kullanılmıřtır; nk Internet'teki ođu zel dokman taranmıř dokmandır. Son olarak veri sızıntısı tespit edilen e-postalar Elasticsearch'de daha sonra bađlamsal analiz iin indekslenmiřtir.

Anahtar Kelimeler : veri sızıntısı, e-posta filtreleme, VSE, ierik analizi

ACKNOWLEDGEMENTS

I would like to express my sincere gratitude to Asst. Prof. Dr. A. Nurdan SARAN for her supervision, special guidance, suggestions, and encouragement through the development of this thesis.

It is a pleasure to express my special thanks to my family for their valuable support.

TABLE OF CONTENTS

STATEMENT OF NON-PLAGIARISM PAGE	iv
ABSTRACT	v
ÖZ	vii
ACKNOWLEDGEMENTS	ix
TABLE OF CONTENTS	x
CHAPTERS:	
STATEMENT OF NON-PLAGIARISM PAGE	iv
ABSTRACT	v
ÖZ	vii
ACKNOWLEDGEMENTS	ix
TABLE OF CONTENTS	x
INTRODUCTION.....	1
1.1 Background.....	1
1.2 Literature.....	2
1.3 Security Concepts	3
1.3 Motivation.....	6
1.4 Objectives	7
1.5 Outline of the Thesis.....	7
DATA LEAKAGE/LOSS PREVENTION	8
2.1 DLP Definition	8
2.2 Type of Data Leakage	9
2.3 Data States	9
2.4 DLP Threats	10
2.4.1 Accidental Data Leakage/Loss.....	11
2.4.2 Insider Threats.....	12
2.4.3 External Threats	12
2.5 DLP Technologies	13
	x

2.5.1 Data Classification	14
2.5.2 Content/Context Analysis	16
2.5.3 Keyword/String Matching.....	18
2.5.4 Regular Expressions.....	19
2.5.5 Statistical.....	20
2.5.6 Fingerprinting.....	20
2.5.7 Machine Learning	20
2.5.8 Conceptual/Lexicon	20
2.5.9 Categories.....	21
2.6 Enforcement Actions	21
2.7 E-mail DLP	22
DESIGN AND IMPLEMENTATION.....	24
3.1 Introduction.....	24
3.2 E-MailRelay.....	24
3.3 Architecture.....	25
3.3.1 E-MailRelay Local Parsing Architecture	26
3.3.2 E-MailRelay Remote Parsing Architecture.....	31
3.3.3 E-MailRelay Multiple Process Architecture	34
EVALUATION.....	36
4.1 Introduction	36
4.2 Test Environment.....	36
4.3 Test Data	38
4.4 Test Methodology	38
4.4.1 Apache JMeter	39
4.4.2 Test scenarios:	40
4.4.3 Thread groups (TG)	41
4.5 Test Results.....	44
4.5.1 Email Relay No Filter Test.....	44
4.5.2 Email Relay Local Parse Test	47
4.5.3 Email Relay Remote Parse Test	50
4.6 Test Evaluation	58

CONCLUSION AND FUTURE WORK	64
5.1 Conclusion	64
5.2 Future Work	65
REFERENCES	66
APPENDICES:	
APPENDIX A: CURRICULUM VITALE	71

LIST OF FIGURES

FIGURES

Figure 1 Components of Information Security (Michael E. Whitman, 2011)	4
Figure 2 High-level Tika Architecture (Chris A. Mattman 2011)	18
Figure 3 E-Mail Relay Local Architecture	26
Figure 4 Email Relay Instance 1 startup command	27
Figure 5 Email Relay Instance 2 startup command	27
Figure 6 EmailRelay DLP Architecture-1	28
Figure 7 E-Mail Relay Filter Script Flow	30
Figure 8 E-Mail Relay Remote Architecture	31
Figure 9 EmailRelay DLP Architecture 2	32
Figure 10 EmailRelay Server 2 Startup Command.....	32
Figure 11 Hardware Setup.....	37
Figure 12 Test plan	39
Figure 13 Email Relay No Filter Test 1	45
Figure 14 Email Relay No Filter Test 2	45
Figure 15 Email Relay No Filter Test 3	46
Figure 16 Email Relay No Filter Average Delay	46
Figure 17 Email Relay Local Parse Test 1	47
Figure 18 Email Relay Local Parse Test 2	48
Figure 19 Email Relay Local Parse Test 3	48
Figure 20 Email Relay Local OCR	49
Figure 21 Email Relay Local Average Delay	49
Figure 22 Email Relay Remote Parse Test 1	50
Figure 23 Email Relay Remote Parse Test 2	51
Figure 24 Email Relay Remote Parse Test 3	52
Figure 25 Email Relay Remote Test 4 OCR	52
Figure 26 Email Relay Remote Parse Average Delay	53
Figure 27 MyDLP No Filter Test 1	54
Figure 28 MyDLP No Filter Test 2	54
Figure 29 MyDLP No Filter Test 3	55
Figure 30 MyDLP No Filter Average Delay	55
Figure 31 MyDLP Test 1	56
Figure 32 MyDLP Test 2	57
Figure 33 MyDLP Test 3	57
Figure 34 MyDLP Average Delay	58
Figure 35 Test Scenario 1/2 Results	59
Figure 36 Test Scenario 3 Results	60
Figure 37 Test Scenario 4 Results	61
Figure 38 Email Relay Filter/No Filter Comparison	62

Figure 39 MyDLP Filter/No Filter Comparison63

LIST OF TABLES

TABLES

Table 1. Type of Information Leaked (Peter Gordon, 2007).....	9
Table 2. Test Environment.....	37
Table 3. Test Data.....	38
Table 4. Test Thread Groups Matrix.....	40

LIST OF ABBREVIATIONS

HIPAA	Health Insurance Portability and Accountability Act
PCI	Payment Card Industry Data Security Standard
PII	Personally Identifiable Information
DLP	Data Leakage/Loss Prevention
OCR	Optical Character Recognition

CHAPTER 1

INTRODUCTION

1.1 Background

Data is the most important asset for corporates. Every application and every server is for serving data. We build applications and deploy these to servers to serve to people over the Internet or over the internal network. Also we secure these servers from the inside to outside; because data is important. Since data is so important; data have to be protected, have to be classified, accesses to it have to be monitored or may be filtered when going to outside.

Not all of the data in corporate is about corporate data, some of them are personal data. There are data regulations and legislative laws for protecting personal data. Some of the legislation is general, and some of them is specific to sector. For example healthcare compliance is required by law throughout the US. In order to accomplish HIPAA compliance, the records for patients have to protected from leaked to third parties via ftp, e-mail or other methods. Another example may be the finance sector. There are also regulations specifying the policies and procedures for governing financial protection which is the most notably PCI standard. Also there may be data security policies in corporates for non personal data which can be classified as trade secret. For example a corporate can support research projects. These project names may be classified as trade secret.

Data leakage or data breach occurs when a party sends the regulated/sensitive data to untrusted third parties. This is an undesirable situation. In order to avoid this

situation, many vendors developed Data Leakage Prevention (DLP) solutions. There are also open source DLP solutions available. DLP solutions help identification, monitoring, protection and reducing of the risks of sensitive data leakage [4]. In addition to avoid data breach, many organisations buy these commercial products/solutions in order to have regulatory compliance.

1.2 Literature

Research in DLP can be classified into abuse detection in information retrieval system, e-mail security, web based security, encryption and access control, hidden data in files, honeypots and honeytokens [32]. Summary of the researches about DLP field can be found in the surveys written by Shabtai and Alneyadi [27,32]. One of these is e-mail leakage which this thesis is about. E-mail leakage is a scenario of highly critical importance because of exposing information accidentally by insecure e-mail communication (i.e., wrong attachments, wrong address, etc.) [27].

There are mainly two approaches which are content-based and behaviour based. These are the approaches which shall detect and prevent data leakages. The content-based approach is further divided into two categories which are keyword-based rules and machine learning techniques. In the keyword-based rule approach, rules are generated from the keywords. The keywords appear in the body and the header of the e-mail [27]. The confidentiality level of the e-mail is determined by the rules which are based on the number of occurrences of certain keywords [34][35][36]. In addition Polatcan investigated the abnormal e-mail activity based on the language of the e-mail and the use of attachments within the e-mail [8]. He developed the Invisible Witness Tool which alerts the administrator if the language of the message is different and the spelling error in the message is greater than 25% [8]. Moreover the other well known content based detection is fingerprinting. Fingerprinting is the conversion of the document into a set of hash values. Fingerprint is extensively used in plagiarism detection. Shapira and Shaptai proposed a method which can generate

fingerprint of the exact confidential content and ignores the non-confidential parts with the help of k-skip-n-grams [29].

In the machine learning techniques approach, the fundamental idea is to use machine learning techniques such as support vector machine method (SVM) [37] or Naive Bayes [38][39][40] method to determine the confidentiality level of the scanned message [27]. Regular expressions, introduced by Kleene [33], is also another method used in DLP systems. Regular expressions are sets of characters or terms that are used to form detection patterns. In information security, regular expressions are used in data filtering, data inspection for malicious codes or confidential data. Becchi and Crowley [41] made the detection of patterns in packet payload faster with using regular expressions along with a compression algorithm [27]. In addition, exact and partial detection of personal information such as social security number, payment card numbers or confidential data in organisations are done with regular expressions [42]. Also using dictionary based techniques may improve detection faster [32].

Stamati-Koromina, Ilioudis, Overill, Georgiadis, and Stamatis proposed an e-mail data leakage prevention system which is able to identify, prevent and log e-mail messages having leak information from an organisation with the aid of steganography [55].

1.3 Security Concepts

Information is defined *as* “facts and ideas, which can be represented (encoded) and processed as various forms of data,” *and* data *as* “information in specific physical or raw representation, usually a sequence of symbols that have meaning; especially a representation of information that can be ready to be processed or produced by a computer.” The two definitions are not much different than other [10].

Preventing unauthorised access to data, data leakage, misuse of data or modification of data is important in information security to provide confidentiality, integrity and

availability [14]. Organisational policy, training, information security awareness and also the technology used is also important to protect the confidentiality, integrity and availability of information assets. Information can be stored in a database file or it can be used interactively or it is already send via network [16].

Computer security is defined as the affordable protection of an information system to reach the acceptable goals of maintaining the integrity, confidentiality and availability of informations resources such as hardware, software, information or data [15].

As seen in Fig. 1 information security is the union of security management, computer security and network security. The computer security industry developed a concept model of information security known as CIA triangle. The security concept is best understood with CIA triangle [16].

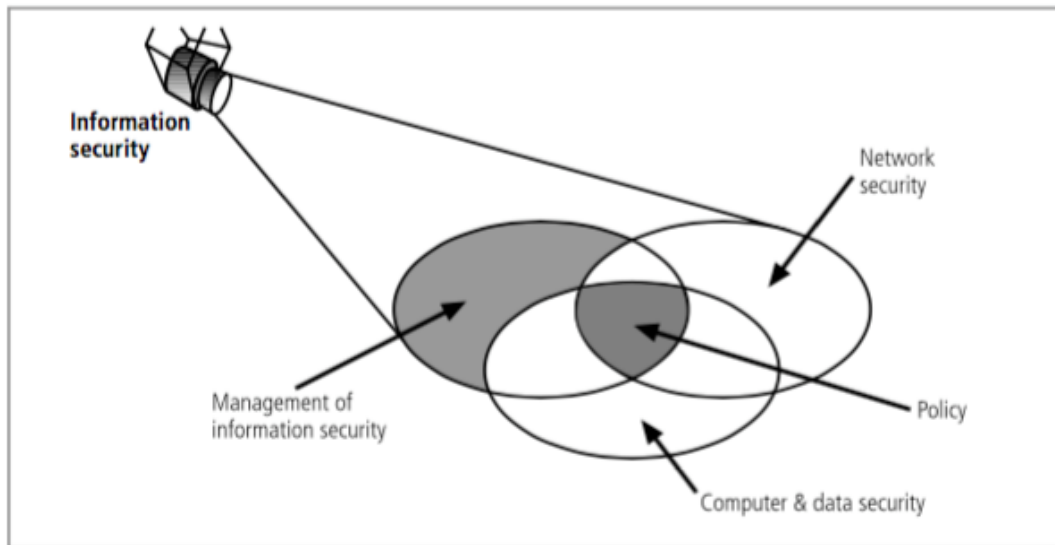


Figure 1 Components of Information Security (Michael E. Whitman, 2011)

The CIA triangle definition can be expanded as follows [10]:

- Confidentiality:
 - Data confidentiality is making the classified information not available to unauthorised parties. One of the best effective method is data encryption [10].

- Privacy is making sure of the scope, storage and disclosure of information needed by the party [10].
- Integrity:
 - Data Integrity is ensuring the change of data in information is defined and authorised [10].
 - System Integrity is ensuring a system free from accidental or intentional unauthorised manipulation of the system and a system which performs its planned function as well [10].
 - Availability is making sure of a available system service to authorised users and users are not denied to use it [10].

Some other key terms and concepts are listed below:

Access: Authorised users can access to a system in a legal manner, on the other hand unauthorised users like hackers do not have legal access to a system. Access controls regulate this ability [16].

Asset: A protected organisational resource which is often critical to organisation. An asset is a logical entity, an information or data such as human resources database. An asset is physical entity such money, employee or a hardware. Assets, especially information assets, are the focus of security attempts in which these are supposed to protect [16].

Attack: An external or internal act that can break into organisation's computer system and cause damage to the system, expose information from the system that support the organisation. Attacks can be passive or active, for example reading a sensitive information by someone that is not supposed to be seen in an unauthorised manner is a passive attack. Accessing an information system in an illegal manner by a hacker is an intentional attack [16].

Exposure: A condition or state of which an information is accessed using a legal or illegal manner. Exposure exists when an attacker can compromise the system with a known vulnerability [16].

Loss: Unauthorised modification or disclosure of organisational information asset. When it's stolen, organisation has suffered a loss [16].

Protection profile: A profile consisting of controls including training, education, policy, and technology that the organisation implements to protect the asset. It is also used interchangeably with the term security program which often consists of managerial aspects of security like planning [16].

Risk: The probability of something unwanted happening multiplied by the resulting cost. Organisations minimise risk to match their risk via risk management [16].

Threat: A possible harmful event that may occur for an vulnerable asset. Threats can be intentional or accidental. For example, hacking is an intentional threat, whereas possibility of a natural disaster like tornado can be accidental [16].

Vulnerability: A weakness condition in absence of some security safeguards that could be attacked/exploited by a threat [15].

1.3 Motivation

Corporates want to be aware of any data leakage if a given keyword or pattern appears in the e-mail content. After an intentional or unintentional data leakage occurs via e-mail leakage vector, an e-mail administrator have to search the keywords using the tools provided by the e-mail server vendor. In most cases, for example forensic investigation cases, administrator have to search the keyword in a large timeline. This process may be time consuming. For this purpose one can use

DLP e-mail system in order to be notified by data leakage or in order to block the leakage or both of them.

According to Europe Data Protection regulation, companies may be punished in case of data breach. In our country if a data breach occurs, it is handled via TCK cybercrime law articles 243-246 [45]. Since personal data is processed, companies have to obey on Turkish Law on Protection of Personal Data no 6698 [46,47].

1.4 Objectives

The objectives of this research are as follows:

- Develop and integrate a system to find leakages via e-mail.
- Develop a scalable system.
- Consider the all parts of the messages, i.e binary attachments.
- Consider the embedded images in the attachments.
- Test the proposed system.

1.5 Outline of the Thesis

This thesis contains five chapters and the remaining of this thesis is organised as follow:

In Chapter 2, general information about some introductory concepts and information about DLP technologies are given.

In Chapter 3, the design and implementation of the architecture is explained. The developed parser scripts with different architectures are given.

In Chapter 4, the E-mail DLP system is evaluated and tested with different scenarios. The performance and the black box tests and comparisons are given in this chapter. Finally in Chapter 5, the future work and conclusion remarks are given.

CHAPTER 2

DATA LEAKAGE/LOSS PREVENTION

In this chapter some introductory concepts and techniques for data leakage/loss prevention technologies will be introduced.

2.1 DLP Definition

The well known security practice is first defining the valuable information assets, then analysing the value, vulnerability and threats among these assets, and finally making risk assessments to protect the assets. In this practice a set of known or unknown threat actors and authorised users are behind the scene [57]. Authorised users sometimes abuse their legitimate authority. When an authorised user send sensitive information to outside whether intentionally or not, an information/data leakage occurs. In order to detect and prevent possible data leakage incidents in a timely manner, Data Leakage/Loss Prevention (DLP) systems are designed. These systems prevent the data leakage incidents by monitoring data which can be in use (endpoint) or in motion (network traffic) or at rest (data storage) [4].

DLP changed considerably which addresses risks that are concentrated on specific class of data in information security. These risks can be personally identifiable information (PII) such as payment cards, banking or legal information. Disclosing this data outside an organisation's security perimeter leads to issues that are harmful to organisation [4].

According to Securosis, a research firm, DLP is defined as "Products that, based on central policies, identify, monitor, and protect data at rest, in motion, and in use, through deep content analysis." [5]

2.2 Type of Data Leakage

To protect the information assets, an awareness on the assets to be protected have to be created. Based on publicly revealed data leakage incidents, the type of data leaked is broken as in Table 1 (Peter Gordon, 2007) [6]:

Type of information leaked	Percentage
Confidential information	%15
Intellectual property	%4
Customer data	%73
Health records	%8

Table 1. Type of Information Leaked (Peter Gordon, 2007)

Also according to another report by Infowatch, the type of data leaked globally in 2016 are %62 of personal information, %31 billing information, %3.9 state (government confidential) secret and %2.8 trade secret and know-how [58].

2.3 Data States

In DLP terminology, there exist three different states of data which are data at rest, data in motion and data in use. DLP solutions are differentiated between these three different data states [4].

The data-at-rest state is for the data which are stored in computer storage. To prevent data from being accessed, stolen, or modified by unauthorised parties, access to the

data is controlled and data can be encrypted to provide confidentiality. In order to use these security measures, one has to do the content discovery [4].

The data-in-use state is for the data which a user interacts with. To protect and monitor the data at the user level, endpoint systems are used. Endpoint systems are agent softwares which are installed on the user client. These agent software monitors the user operations while the user processes the data or transferring the data from one medium to another. The data-in-use DLP agent software can monitor the following user activities:

- Screen capture and copy-paste operations involving sensitive data.
- Transferring of sensitive content from one channel to another using electronic, magnetic, or optical storage devices such as USB drives, CD/DVD, smart phones, and PDAs.
- Printing, or faxing sensitive content [4].

The data-in-motion state is for the live data which are sent inside the internal network of the organisation or sent to outside network of organisation across active/passive network devices. DLP solutions in this category examine the known protocol (SMTP, HTTP, IM) data or unknown protocol data (packet payload) which are sent across the network. On the other hand, a DLP solution becomes worthless if encrypted data can not be decrypted at the gateway security device or encrypted data is permitted without being able to decrypt first. Because DLP solution can not examine the content of the data in motion [4].

2.4 DLP Threats

According to data breach incident report compiled by Verizon, 77% of insider and privilege misuse category is from internal actors and 11% of it is from external actors [59].

Most of the internal leakages are from unplanned actions like employee failure or problems in business processes. This kind of problems are difficult to solve, and it is not so easy like installing a secure reverse proxy server. Mostly before installing a DLP solution, a sensitivity assessment or inventory of business processes have to done. In a sensitivity assessment both the information and system addressed. It includes legal connections, organisation policy, and the functional requirements of the system. Since sensitivity is normally indicated with confidentiality, integrity and availability, the outcomes of unauthorised disclosure, unauthorised modification or the data need to be considered while assessing sensitivity [15]. Business processes may be changed or re-engineered. Personnel training on information leakage or information security policies have to done. After that, a cultural change is expected within the organisation [6].

Although the malicious internal leakages is remarkably low, there is still a risk of intentional unauthorised disclosure of information by insiders. The leakage channels are remote access, e-mail communication, instant messaging, peer to peer communication, file transfer protocol or portable USB disks [6].

2.4.1 Accidental Data Leakage/Loss

A threat may be a result of an accident, such as employee incorrectly entering information on a system, which results in the system malfunctioning [10].

A typical cause for accidental data leakage is that the policies are not known by employees. That is to say, there is not any classification on documents or not any policy to work with. In addition they do not have any computer security training. Therefore the employees do not know the sensitivity of the documents. Some common accidental data leakage scenarios are:

- Sending sensitive documents unencrypted via e-mail.
- Saving documents to a USB storage device for further transportation.

- Uploading files to online services. [7]

2.4.2 Insider Threats

Insiders are called the authorised users of a system. Insiders are current or former employees, and partners who have authorised access to the information system. Although insider threats are in a better place to do illegal or abnormal activities, an unplanned disclosure shall be done by insiders without a target or malicious motive [9]. On the other hand, employees with high technical background can do malicious activities such as breaking an information system without an authorisation.

Attacks from the organisation inside are the most valuable information security incident. The average cost per insider is £250,000 according to a report by the INSA[9]. Insider attacks are the most expensive form of information security breach, in that the average cost per insider incident is £250,000 according to a recent report by the INSA[9].

The most common types of technologies used by organisations to concern with insider threat problem cover web log analysis in firewall or proxy servers, log analysis in audit systems, forensic analysis after incident, messaging and communication analysis. Each of these technologies are necessary security measures implemented in IT organisations to address point or areas of vulnerabilities in corporate networks and computer assets [13].

2.4.3 External Threats

An internal threat is a threat which develops inside the organisation. In contrast external threat is a threat which develops outside the organisation. An external threat gain unauthorised access to the internal private network of an organisation with social engineering, malicious software, spam, phishing, etc. Also making the

organisational resources unresponsive with flooding the network with large volumes of requests is also another example of external threat. Also most of these threats are intentional.

External threats can be blocked by securing the perimeter of the network with firewall systems. Firewalls are systems located between the external network and the internal network to selectively pass or filter traffic based on policies [11].

Some external threats in the threat community are:

- Cyber-criminals (professional hackers)
- Spies
- Non-professional hackers
- Activists
- Nation-state intelligence services (e.g., counterparts to the CIA, etc.)
- Malware (virus/worm/etc.) authors [12]

Moreover to protect TCP/ IP networks from external threats are intrusion detection/ prevention systems (IDS/IPS), anti-virus protection and anti-spam technology. Intrusion detection is simply determining unauthorised use of computers and networks [57].

2.5 DLP Technologies

Data Leakage Prevention (DLP) technologies is designed to protect sensitive personel or confidential corporate data. Sensitive information in various leakage channels such as e-mail, instant messaging, portable USB drives can be monitored and protected [28].

The DLP technologies is divided into DLP as a feature, and DLP as a solution. A number of products such as next generation firewalls and e-mail security solutions

have lite DLP features, but they are not complete DLP solutions. A DLP solution contains centralised management, policy creation, and enforcement (block, accept, audit) workflow. It is dedicated to monitoring and protection of content and data [28].

As previously mentioned in section 2.3 there were three states of data which are data-at-rest, data-in-use and data-in-motion. DLP technologies arise from the state of data. There are two types of DLP technologies which are endpoint DLP and network DLP while former arise from the data-at-rest and data-in-use, the latter arise from data-in-motion. So e-mail DLP fits in the data-in-motion type. While endpoint DLP technologies runs on the client PC as an agent which gets the policies from a central DLP server. For network DLP, there is no requirement for an agent.

In endpoints where DLP agents installed, for example copy/paste operation can be done, but transparently sent encrypted to the third parties without the user knowing. While network DLP systems have predefined compliance templates, sometimes it's not enough to have predefined templates, also it is needed to have auto learn and tagging features. Since sometimes end user don't want to tag data-in-use content for classification purposes.

The technologies used in DLP are in the below sections.

2.5.1 Data Classification

Data classification can be defined as a tool for categorisation to help organisation to answer the following questions [18]:

- What data types are available?
- Where are certain data located?
- What access levels are implemented?
- What protection level is implemented and does it adhere compliance regulations?

Data classification, also known as information extraction in the field of information retrieval, is the process of extracting semantics/meaning from text. The main motivation is the need to protect and control the use of sensitive information within the enterprise [21].

A variety of classification schemes, which are division of individual objects into classes or groups having common characteristics, are used by corporates and military organisations. To secure the confidentiality of information, data classification schemes are used by many corporations. The typical classification scheme has three categories which are confidential, internal and external. Information owners are responsible for classifying the information that they are responsible. Periodically information owners must review the classifications to ensure the correctness of classification and the appropriate access controls are made [16].

Confidential classification is used for the most sensitive corporate information that must be controlled within the company. Access to confidential information is based on a need-to-know basis or as required by terms of a contract. Information with this classification also referred as sensitive or proprietary [16].

Internal classification is used for all information which are internal to the company and is to be viewed only by corporate employees, authorised contractors, and other third parties [16]. External classification is for public information which are approved by management.

Document or data classification is not only done by hand or manually. There are also algorithmic classification methods which are mainly in information science and computer science.

2.5.2 Content/Context Analysis

There are three approaches:

- Context-based inspection
- Content-based inspection
- Content tagging

All of the security technologies such as spam filters, proxies, firewalls, and intrusion detection/prevention systems (IDS/IPS) uses context-based inspection methodologies. The term context consist of information extracted from monitored application data such as source, destination, sender, recipients, metadata information, time stamps, file location, file type, etc.

Packet-filter firewall is an example of context inspection-based system. It decides whether a network packet will be allowed to pass through with the help of some policy filter rules such as source/destination IP address, application type, destination port. Moreover a context-based DLP solution can prevent java source code files from being send out of an organisation, block all encrypted files or prevent copy/paste operations from specific applications [27].

Data leakage can be detected with content-based inspection which analyses content with a variety of methods such as keyword matching, fingerprinting, statistical and machine learning techniques.

In addition to content-based approach, a file containing sensitive data is tagged and a policy is applied to the assigned tag. Assigned tag remains in the content also when other applications processes. Tags can be assigned to files manually (by the creator of the sensitive data) or automatically (using content/context based analysis, based on file location or based on file creator applications/users).

When dealing with content that is not plain text, such as binary files, it is important to look several levels down to a file. For example when an Excel spreadsheet is embedded in a Word file, the content in the spreadsheet file have to be analysed. So a DLP solution should provide this kind of "file-drilling" capabilities to interpret a file.

For this purpose the Apache Tika project can be used for detecting content and meta data of binary files. Apache Tika toolkit is a content detection and analysis framework written in Java language. It detects and extracts texts and metadata from many different types of files. The toolkit also provides a Java library, server and command line versions. It is also suitable for use from other programming languages [22].

Tika's key components are shown below:

- Parser framework
- MIME detection mechanism
- Language detection
- Component that ties all of the components together
- External interfaces including command line and graphical user interface

The high level Tika architecture is shown in Fig. 2 (Chris A. Mattman 2011).

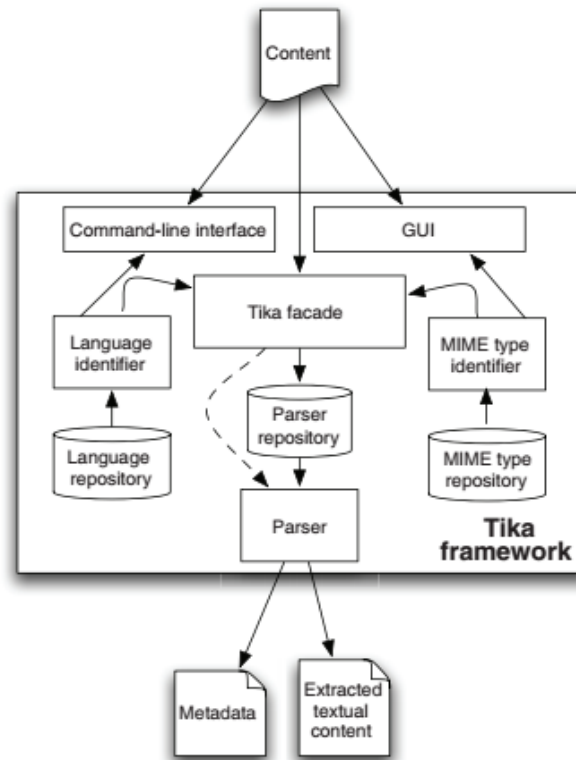


Figure 2 High-level Tika Architecture (Chris A. Mattman 2011)

In this work, Elasticsearch is used for auditing context of data, i.e the matches from the data. Elasticsearch is an open-source search engine built on top of Apache Lucene which is a complex full-text search engine library. Elasticsearch uses Apache Lucene library internally and make the search easier and distributed with restful APIs. Elasticsearch scales well since it is distributed. In addition to full-text search it is used for analytics also. Data analytics can be done with the Kibana server. As a result Elasticsearch is a distributed real time document store where every field can be indexed and analysed. It also provides real time analytics and hundreds of servers and petabytes of structured and unstructured data can be scaled.

2.5.3 Keyword/String Matching

One of the simplest representation of information is called a text which is characterised by a long linear sequence of characters. The texts are processed and used with the help of text algorithms mainly in every field of science and information

processing. In Unix where most of the information is exchanged with files, in data processing, text editing, symbol manipulation, lexical analysis, code generation, spelling correction, text retrieval and natural language processing the pattern/string matching occurs [25][53].

The basic textual problem is pattern or string matching. It is a frequently used operation in some application system. For example "grep pattern file.txt" command outputs the lines which contains string "pattern" in the file.txt file. Single keyword matching means that all occurrences of a given pattern in the input text string are located [25][53].

So simple keyword matching is appropriate when a small known keywords can identify private data. For example, medical or financial records are within the scope of this criteria [24].

2.5.4 Regular Expressions

There is sometimes no specification of the pattern to find in text. One have to do substring search with some part of the pattern known. Describing the patterns in this scenario is important. A generalised pattern language has developed over the years. It is expressive and powerful for a wide variety of uses. There are different implementations and uses by programs, but in general, this powerful pattern language and the patterns themselves are called regular expressions [26].

Regular expressions describes a notation to specify a set of possible input strings.

The structured datas like social security numbers, telephone numbers, addresses are recognised primarily by regular expressions [24].

Regular expressions can be divided to two types of characters. The special characters like wildcard character are called meta characters, while the rest are called literal or normal text characters.

2.5.5 Statistical

Statistical metrics like frequency of terms are extracted from the content under inspection. Machine learning methods similar to those used for blocking spam are used in this approach. When a deterministic technique is difficult to implement, this statistical metrics approach is efficient for detecting unstructured content [27].

2.5.6 Fingerprinting

A fingerprint is a unique hash value associated with a set of data. Fingerprinting is a method which extracts "fingerprints" from sensitive files or database records. Hash values for sensitive files are stored in databases to detect the data leakages. The fingerprints are searched in the database or locally on the machine under inspection [27].

2.5.7 Machine Learning

In machine learning, on the basis of training set of observed and categorised data, classification is the problem of identifying the category of a new observation. An example could be labelling a given e-mail into "spam" or "non-spam" class/category. Moreover classification is an example of pattern recognition. Also, when there is observed training data, classification is considered an instance of supervised learning. In contrast to supervised learning, there is unsupervised learning known as clustering where there is grouping of data based on similarity of distance on data [20].

2.5.8 Conceptual/Lexicon

Another approach to detect data leakage is by analysing content using lexicons. Lexicon is a list of terms relating to a particular subject. The keywords such as confidential, project 2501, financial report, and patterns or regular expressions

matching can be used to form a dictionary. Most products include common dictionaries that address laws and regulations like PCI-DSS, GLBA, HIPAA and SOX. With this type of configuration it is easy to detect and configure, but it provides little protection in the case of unstructured data [27].

This method combines word usage patterns commonly associated with specific concepts such as critiquing co-workers or doing online shopping. When the detected word and expression count associated with such a concept is high, an alert is triggered [7].

2.5.9 Categories

Categories or DLP policies are the compliance templates which use any classifier or method (word list, regular expressions, etc.) to detect certain types of content. When one of the following PCI-DS category conditions are met, the file or message considered is sensitive:

- 5 credit numbers
- 1 credit number + 1 name
- 1 credit number + 1 partial date + expiry date keywords.

In addition to PCI-DS, templates for other compliance requirements such as HIPASS are also integrated in DLP products. The HIPAA includes common names and personnel ID numbers in American health insurance documents. There are also templates for detecting IBAN, SWIFT/BIC and other more international types of sensitive data which can save a a lot of time [7].

2.6 Enforcement Actions

DLP tool take enforcement actions ranging from alerts to active protection, or some combination [54].

Actions for an e-mail leakage can be:

- Alert/Audit
- Notify administrator
- Move/Quarantine message
- Reject/block message
- Modify message

2.7 E-mail DLP

E-mail Data Leakage Prevention (DLP) is content level scanning of e-mail text messages and attachments embedded in the message to detect unwanted transport of sensitive informations. Personal identifiers or corporate internal or confidential documents are examples of sensitive content [30].

There are many products supporting data leakage/loss prevention for outbound email. Some of them can be named as lite DLP which are software modules for existing security software. There are also new trends in email DLP which are may be named as DLP as a Service for Email. Symantec Data Loss Prevention Cloud Service for Email moves the company's mail to cloud for comprehensive email protection and guaranteed delivery. Gmail data loss prevention (DLP) also lets the organisation's inbound and outbound email traffic for content such as personal identifiable numbers and sets up policy based enforcement actions when this content is detected. Also McAfee Email Protection and Trend Micro InterScan Messaging Security (IMSVa) solutions have built-in DLP technology which have builtin compliance templates (categories), PCI-DSS, healthcare, financial and regional privacy regulations, regular expression patterns, keyword dictionaries matching sensitive data. Open source solutions like MyDLP can also attract attention. Some of the propriety solutions use postfix as a mail server as in Trend Micro IMSVa. The same is true for MyDLP; but MyDLP uses it own content filter server which was

mostly written in Erlang language because of its performance and concurrent network operations [7].

CHAPTER 3

DESIGN AND IMPLEMENTATION

3.1 Introduction

This chapter describes the design and integration of the DLP system used for e-mails. Firstly e-mail relay architecture will be introduced, then different parsing architectures used will be presented. Advantages and disadvantages of those will also be discussed. Finally the reasons for selecting it will be discussed.

3.2 E-MailRelay

In this thesis, in order to find data leakage via email, store-and-forward e-mail server with filtering capability is used. The open source solution named as E-MailRelay SMTP proxy and relay server which is written by Graeme Walker is used.

E-MailRelay is a store-and-forward message transfer agent and proxy server [1]. The server process runs in the background. It can be run in two modes. One of these is storage daemon part waiting for an incoming mail messages and then store these in the spool folder. The other one is a forward agent part pulling the messages out from spool folder and forwards these to the appropriate e-mail server. When email relay server runs in server mode, it stores the messages in the spool directory, process the e-mail messages and then forward these to the appropriate server if forward-to option is given.

In this work, two e-mail relay instance with the poll option is preferred. Because, if one use the same server for inbound and outbound mail, the sender have to wait for

the e-mail relay server to accept and process the message. So in order to avoid this delay, an incoming queue e-mail relay server and an outgoing filter e-mail relay server is used. Email relay server 1 (incoming) as seen in Fig. 5 has been started as server mode. E-mail relay server 2 (outgoing) as seen in Fig. 5 has been started as proxy mode. After email pre-processor runs the parsing filter script, proxy server forwards the message according to the return value of the script.

Incoming messages are stored in the message store which is `/var/spool/emailrelay` by default. For the incoming and outgoing email relay instances, the spool folder locations are different. Email relay server stores the incoming mail message as text file in the message store. Each message is represented as an envelope file and a content file. The envelope file contains parameters relevant to SMTP dialogue. The content file contains RFC822 headers and body text. Email relay server saves the state of the messages via file suffixes:

- `.new` (while the envelope is first being written)
- `.busy` (while the message is being forwarded, a kind of locking mechanism)
- `.bad` (if the message can not be forwarded)
- `.local` (for copies of the envelope file for delivery to local recipients) [1]

3.3 Architecture

In order to find leakage via email data in motion, a topology must be chosen. One of the topology is setting a proxy smtp server between the end user and the actual e-mail server. Also the system administrator can set the proxy server as relay outbound server after the actual email server. In this thesis, the former topology as seen in Fig. 3 is preferred. In this topology, with the help of E-MailRelay server, messages send to server by smtp client like Thunderbird, is stored as a file by E-MailRelay server and parsed via a filter script called by the local or network pre-processor program. After the parsing phase, message can be searched for any keywords or patterns matching the DLP policy given in that script.

3.3.1 E-MailRelay Local Parsing Architecture

3.3.1.1 General Work Flow

In this architecture, the filter script and the e-mail relay server is on the same server as seen in Fig. 3. There are two e-mail relay instances in this architecture. First instance is the incoming e-mail relay server and the second one which the filter script run is the outgoing e-mail relay server.

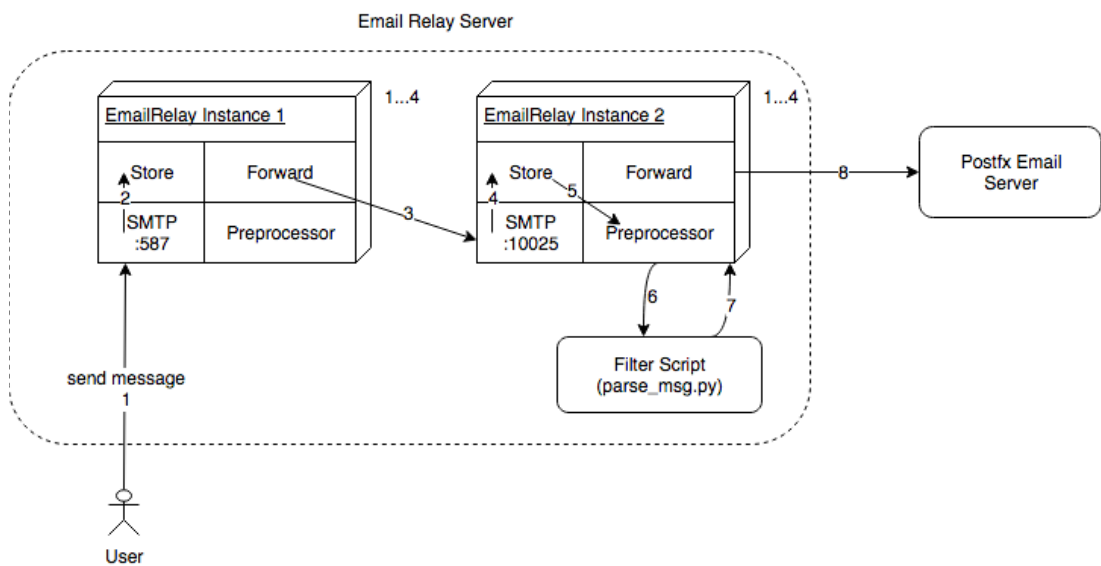


Figure 3 E-Mail Relay Local Architecture

The two e-mail relay instances start from the command line with options as seen in Fig 4. and Fig 5. When a user sends a message to the e-mail relay instance 1, the message is spooled. After the message is being spooled, mail processing starts as in Fig 3. For every given polling time, the e-mails stored in spool folder/store are forwarded. After the incoming message is forwarded to e-mail relay instance 2, the email relay server pre-processor in e-mail relay instance 2 triggers the filter script (`parse_msg.py`) which operates on the message as seen in Fig. 3.

```
/usr/sbin/emailrelay --as-server --remotel-clients --spool-dir /var/spool/emailrelay.in --user emailrelay --client-tls --tls-config=1 --server-tls /etc/pki/tls/mail.pem --forward --forward-to 192.168.122.158:10025 --port 587 --interface 192.168.122.158 --verbose --poll 5
```

Figure 4 Email Relay Instance 1 startup command

```
/usr/sbin/emailrelay --as-proxy mail.test.local:25 --spool-dir /var/spool/emailrelay.out --user emailrelay --forward --forward-to mail.test.local:25 --client-tls --tls-config=1 --log-time --port 10025 --interface 192.168.122.158 --verbose --filter "/home/emailrelay/parse_msg.py" --debug
```

Figure 5 Email Relay Instance 2 startup command

When a user sends a message to the e-mail server as seen in Fig. 3 (step 1), the e-mail relay server processes the incoming message and then forwards the message to the second e-mail relay server for every polling time (step 3). With separating the incoming and outgoing mail, the waiting time of preprocessing of message and returning to the client from e-mail relay instance 1 is avoided. Also while the filter script is running, e-mail relay server 2 is in blocked state.

The general architecture of email relay DLP is given in the Fig. 6, e-mail relay instance 1 is storing the message for buffering and forwarding to the e-mail relay instance 2. E-mail relay instance 2 is calling the simple filter script which was written in Python programming language. So the flow delegates to filter script. Filter script then parses and processes the spooled messages before delivery. The script not only check keywords in the message body, but also the attachments. Apache Tika content analysis framework is used for this purpose. The script transmits the attachment to the Tika server via a REST call and the content in text format is received. Then filter script returns the exit code to the e-mail relay server. According to the exit code, the e-mail message can be forwarded or not. For example, if the script terminates with an exit code of zero, it is a success and processing continues. If it terminates with a value between 1 and 99, it is a failure and processing stops for

the message. Otherwise, for a value of 100, all further processing stops and e-mail relay system does not care the message.

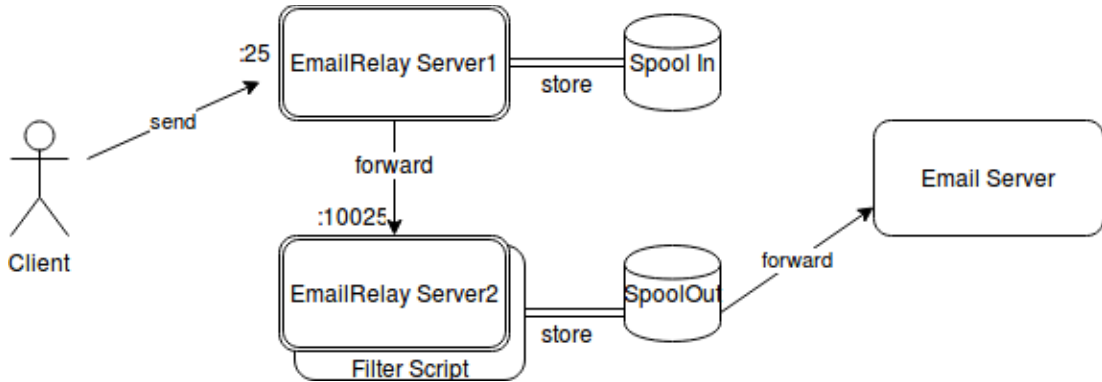


Figure 6 EmailRelay DLP Architecture-1

3.3.1.2 Filter Script

The filter script is written in Python3 programming language. Since Python programming language is widely common and supported by a large community, Python language is preferred. The script gets the message file in the main method and creates a message object structure tree from the given open file object, the flowchart of the filter script is given in Fig. 7. This message tree object passes to a function which walks every part of the message. While walking the message parts, the payload of the message or attachment file is appended to a content list data structure. If there is a binary/app-type content (i.e office file, pdf file, etc...) or if the message has an attachment, the script sends the attachment to the Apache Tika server. Then it is parsed with Apache Tika server from buffer. The script uses the tika-python [2] python module to accomplish this. Also, if there is no attachment in the file, it turns back into filter script. If there is an embedded image on the PDF file, the PDF content can not be parsed with Apache Tika server. Since image parsing is not turned on for performance reasons. The embedded image in the PDF file is extracted and send to a Tesseract OCR API Server. The image is processed by the

Tesseract library [43] and the text is returned. The filter script make a HTTP request to a Tesseract OCR API server. The OCR API server used in this thesis is from the realpython web page [50]. For each scanned PDF file, open source Tesseract OCR engine is used to extract the textual contents. Also some of the Libre Office documents can not be parsed with Apache Tika, so converting these to a PDF document solved the exception.

The textual content received from Apache Tika or OCR server or the text message itself passes to a matching function which searches for each of the given patterns. If there is a match with the given keyword or regular expression, the content and the metadata is saved to the Elasticsearch [23] server for further investigations. After the content analysis, the script returns the exit code to the email relay server 2. If there is a failure, i.e exit 1-99 code from the pre-processor program, the message could be forwarded to the e-mail server. If the exit code is 0, e-mail relay server forwards the message to the e-mail server. If the exit code is 100, then the further message processing for the message is cancelled. The script does not block outgoing message, but store the message according to policy. But it can also be configured to block the matched message. E-mail relay server also has the ability to send a failure reason the the SMTP client (user) if the exit code from the pre-processor program is not zero. Then the pre-processor program searches the first few thousand characters of the message for a "<< or [" followed by ">> or]]" pattern. The text between these pattern is returned to the smtp client.

Since Elasticsearch is used for auditing messages, it is important to harden the security of Elasticsearch server. Because some of the e-mail messages will be stored in the Elasticsearch server, it is important to secure access to this system such as using Iptables firewall or Nginx reverse proxy or Elasticsearch native security products.

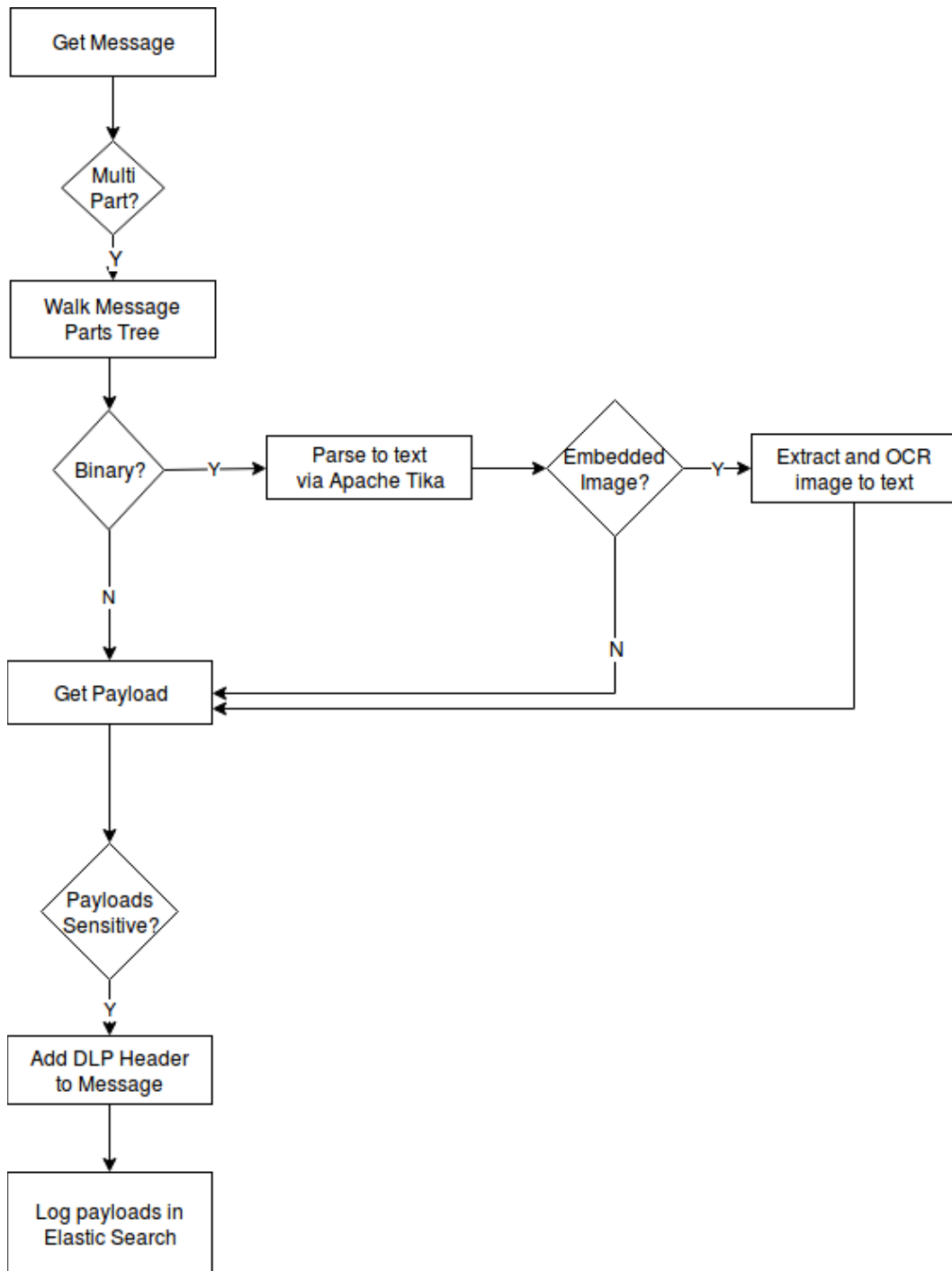


Figure 7 E-Mail Relay Filter Script Flow

The disadvantage of using this architecture is the blocking of e-mail relay system while the filter script is running. There is not any scalability regarding the parse script, but more than one e-mail relay instance can be used to accomplish

scalability. Also it is good practice to have time out for the running script to avoid any dead-lock situation. Also for higher mail throughputs, file I/O waits for filtering will be a problem. On the contrary, if delay times of the messages is not so important, then this architecture is simple to implement. In the following section, another approach to deal with blocking of email relay has given.

3.3.2 E-MailRelay Remote Parsing Architecture

3.3.2.1 General Work Flow

In this architecture, the filter script and the e-mail relay server is not on the same server as seen in Fig. 8. There are e-mail relay instances in this architecture as in local architecture, moreover, a parser server . First instance is the incoming e-mail relay server and the second one which the filter script run is the outgoing e-mail relay server as in local architecture.

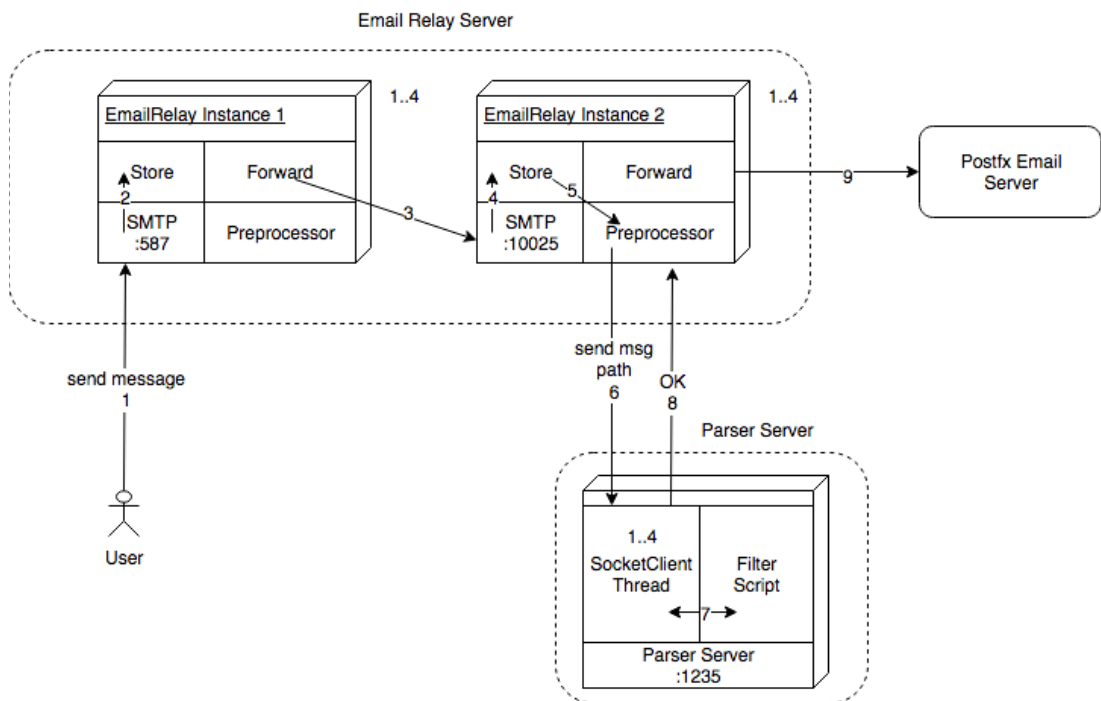


Figure 8 E-Mail Relay Remote Architecture

The architecture seen in Fig. 3 has a problem which is the blocking state of outgoing e-mail relay server while the filter script is running. So a better architecture is implemented as seen in Fig. 8 and Fig. 9.

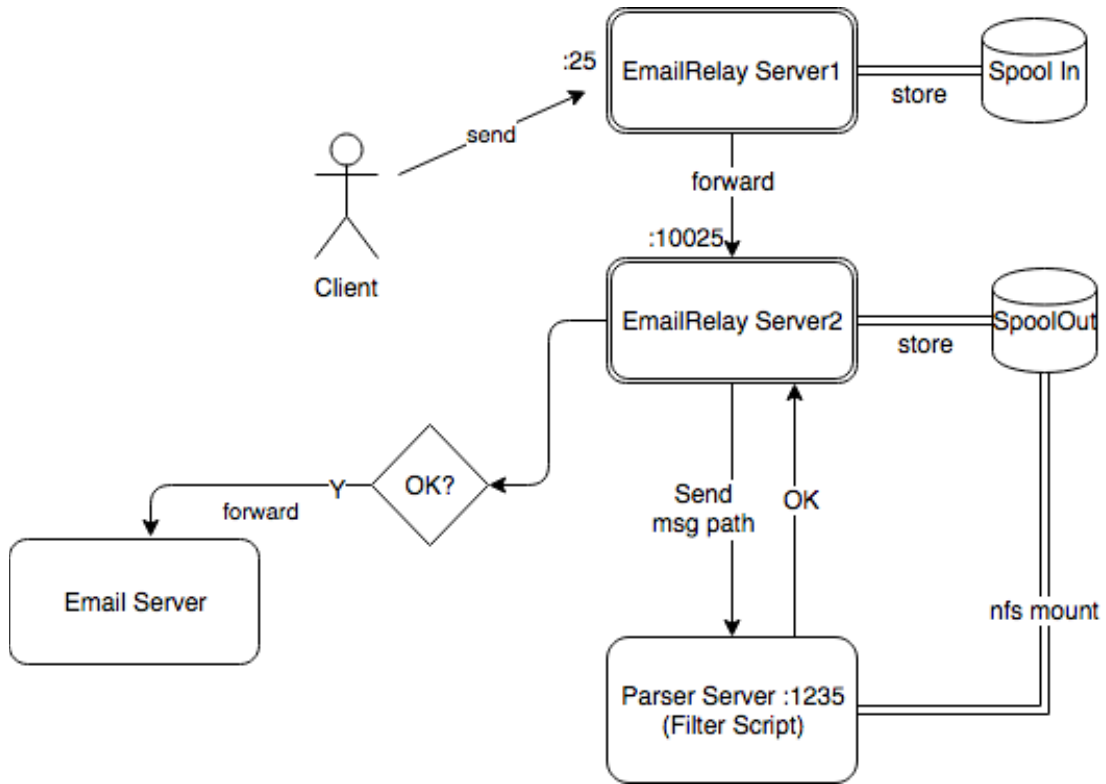


Figure 9 EmailRelay DLP Architecture 2

In this architecture, there is also the same number of email relay instances. In addition to previous architecture, there is a parser server service in which the filtering script runs. The second outgoing e-mail relay instance startup command is shown in Fig 10.

```

/usr/sbin/emailrelay --as-proxy mail.test.local:25
--spool-dir /var/spool/emailrelay.out --user emailrelay
--forward --forward-to mail.test.local:25 --client-tls --tls-
config=1 --log-time --port 10025 --interface 192.168.122.158 --
verbose --filter net:parser.test.local:1235 --filter-timeout 50
--debug
  
```

Figure 10 EmailRelay Server 2 Startup Command

3.3.2.2 Filter Script

When a user sends an e-mail, the e-mail relayed to the second e-mail relay server as before. After the second e-mail relay server gets the message, it connects to the parser server socket and delivers the file path of the message to the parser server process, parser.test.local as show in Fig. 8 and Fig. 9, and then communicates it with using a simple line-based dialog as each e-mail message is processed. As opposed to the previous architecture, the parser server script returns an "OK\n" string instead of "EXIT 0" to accept forwarding of message. The filter is is the same as local parse script except the return values. In the remote parse script, there is "return 0" instead of "sys.exit(0)". The parsing script returns 0 or 100 string to the calling parser server script. For these return values, the parser server returns "OK" to the e-mail relay server. Anything other than "OK\n" is the error message appears in the debug log.

After getting the e-mail message file path from the e-mail relay server, the parsing script is called by the parser server script. The parsing script changes a little in order to interoperate with the parser server script. Then script will have to access the e-mail message file which is on the second e-mail relay outbound spool. At first, the method of getting and putting the file via http call was thought, for simplicity this is not used. In order to process the file, a simple NFS server and NFS client used such that the file message can be processed as if it is on the local system. After pre-processing, the result is being returned to the e-mail relay server.

If the PDF file can not be parsed with Apache Tika server, then there must be an embedded image inside the PDF file. The PDF file's embedded images are extracted with `pdftimages [51]` command into a directory. And to process the images, an http server is running to serve these images from http protocol. The filter script sends the file names to the the Tesseract OCR api server. After getting the file names, the OCR server processes the images remotely.

The advantages of using this architecture is the full scalability. Every service can be scaled which every system administrator wants. Also the e-mail relay server is not blocked while the messages are being pre-processed [3].

The filter script source code can be seen in the author's GitHub repository [60].

3.3.2.3 Parser Server Script

For the second architecture, a server script which binds to given socket and listens is written with Python programming language. It runs in forever loop after opening the socket and waits for the incoming line of text data. The server uses the threading module in python. A socket thread per e-mail relay server client is created for an accepted incoming connection. If there is a connection, the data returned from e-mail relay server is used to open the e-mail message file. After opening the message, the parsing script call starts. If the return from parse script is 0 or 100, a line of 'OK' with carriage return is returned to the e-mail relay server so that the message will be forwarded.

The parser server code interact with not only one e-mail relay server. It can interact with many e-mail relay servers with the help of threads. Thread based parallelism is used in the parser server code. If threads do not used and there are more than one e-mail relay server connecting to the parser server, while the parser server is communicating with a e-mail relay server, the other ones have to wait for the communication to close. So threads are used, if there is any necessary for multiple outgoing e-mail relay servers and a one parser server.

3.3.3 E-MailRelay Multiple Process Architecture

E-mail Relay is a single threaded server. E-mail Relay server uses non-blocking asynchronous network i/o, with a select() event loop. This event model means that the server can handle multiple clients simultaneously from a single thread and the

only significant blocking occurs when external programs are executed [52]. So e-mail relay local parsing architecture is expected to have blocking issues.

If the filter script is slow, then one should use multiple e-mail relay instances/processes running against the same spool directory to regain parallelism. Moreover for higher loads sometimes one e-mail relay instance is not enough to consume the incoming messages. Since e-mail relay server has file locking (mutex) mechanism, more than one e-mail relay process can be run on the same/shared spool directory. So the e-mail relay instance number can be increased. The inbound instances have common spool directory and outbound instance have common spool directory so that the mails in directory are consumed faster than without multi e-mail relay instances. So using multi e-mail relay instances is beneficial.

When using multiple e-mail relay instances, the e-mail relay server benefits from the threaded parsing server in e-mail relay remote parsing architecture. Since the parser server create a socket client thread per client, when using four e-mail relay instance, four thread can be created if needed.

CHAPTER 4

EVALUATION

4.1 Introduction

In this chapter the testing phase of the proposed system is presented. The test frameworks and used methodologies are discussed. In addition to testing, a kind of load testing, benchmarking results are also presented.

4.2 Test Environment

A local test environment is set up with 6 virtual server machines. All of the server and applications are running in Linux operating system. CentOS 7 x64 minimal is chosen. Server operating systems are virtualised with KVM software. The hardware setup is on a laptop with configuration of 4 GB RAM, 1.3 GHz Dual Core and a 200 GB SSD disk and a physical machine with 8 GB RAM in which Apache Tika and OCR run as seen in Fig.11. The virtual machine resources are shown in Table 2.

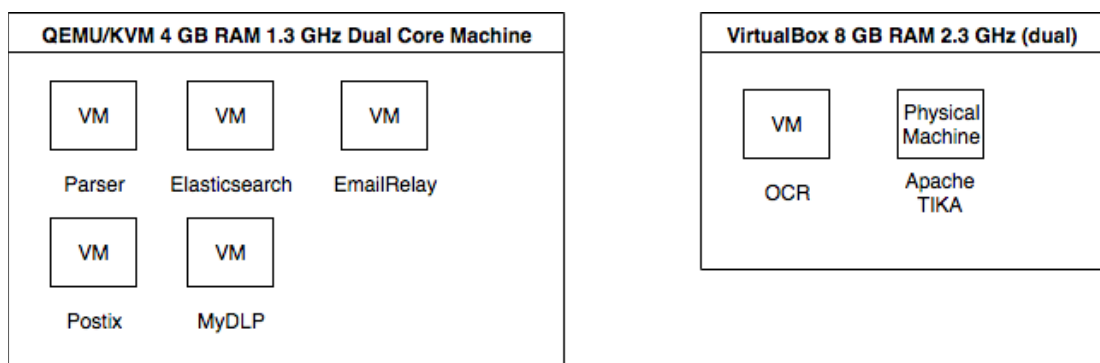


Figure 11 Hardware Setup

<ul style="list-style-type: none"> • Bind DNS/Parse Server <ul style="list-style-type: none"> • 1 vCPU (1.3 GHz) • 384 MB RAM 	<ul style="list-style-type: none"> • ElasticSearch Server (v5) <ul style="list-style-type: none"> • 1 vCPU (1.3 GHz) • 768 MB RAM
<ul style="list-style-type: none"> • EmailRelay Server (v1.9) <ul style="list-style-type: none"> • 1 vCPU (1.3 GHz) • 512 MB RAM 	<ul style="list-style-type: none"> • Postfix Server (v2.10) <ul style="list-style-type: none"> • 1 vCPU (1.3 GHz) • 384 MB RAM
<ul style="list-style-type: none"> • MyDLP Server Appliance (v3.2.0) <ul style="list-style-type: none"> • 2 vCPU (1.3 GHz) • 1024 GB RAM 	<ul style="list-style-type: none"> • Apache Tika Rest Server <ul style="list-style-type: none"> • 2 cores 2.3 GHz • 8 GB RAM (On physical machine)
<ul style="list-style-type: none"> • Tesseract OCR on Flask Server VM <ul style="list-style-type: none"> • 2 vCPU (2.3 GHz) • 2046 MB RAM 	

Table 2. Test Environment

Bind DNS server is required for the test system; because DNS resource records which are MX and A records are required to test send and receive e-mails. Since most of the corporates have e-mail servers, they also have DNS servers in order to give e-mail service [44]. Elasticsearch server is used for logging and analysing the data. Parser server is required for email relay remote tests. EmailRelay is required for getting the SMTP mail message and detecting data leakages. Postfix is used as a production e-mail server. MyDLP server appliance is used for comparison. Apache Tika Rest server is used for getting text content from binary input. OCR server is used for getting text from the embedded images.

The e-mail relay server has 4 incoming instance/process and 4 outgoing instance/process.

4.3 Test Data

Different test data is prepared for different tests. For sending text messages, the author's real messages are used. For sending messages with attachments, the files downloaded from the Internet used. Test data is summarised in Table 3. The false positive non-leak document has an TC-ID number according to algorithm, but according to the context it is not TC-ID number. Here, false positive is when a normal 11 digit TC-ID pattern is thought to be TC-ID number as seen in Table 3.

Test Type	Sample Count	Sensitive Count	False Positive Count
Text Messages	120	0	0
Sensitive Text Messages	120	19	0
Attachment Files	120 (non-ocr)	16	1
Attachment Files with OCR	120 (5 ocr, 115 non-ocr)	21	1

Table 3. Test Data

4.4 Test Methodology

The components of the system have been tested separately. First of all, to parse the incoming e-mail message file, a test script has been written. Batch (offline) testing of 170000 emails are parsed successfully with the script.

In order to compare the results of proposed system, an another open source DLP software, MyDLP (smtp module) has been used. In order to test different testing configurations, the outgoing email relay server's startup scripts and the filter script are customised.

Tests types are:

- Stress/Load Testing of e-mail relay server without filter
- System and load testing of e-mail relay server with local parsing
- System and load testing of e-mail relay server with remote parsing
- System and load testing of MyDLP server with filter
- System and load testing of MyDLP server without filter

In order to test the stability and scalability of the system, load testing of the email relay server is done.

4.4.1 Apache JMeter

Apache JMeter has been used as a testing tool, as shown in Fig. 12. The JMeter GUI is only used for changing the test plan. The tests are started in command line. For sending mails JMeter SMTP Sampler is used. For reading mails, the JMeter Mail Reading Sampler with post bean shell sampler has been used. Also bean shell java scripting language is used in various thread groups such as message sending, raising assertions, etc. The test codes can be seen in the author's GitHub repository [60].

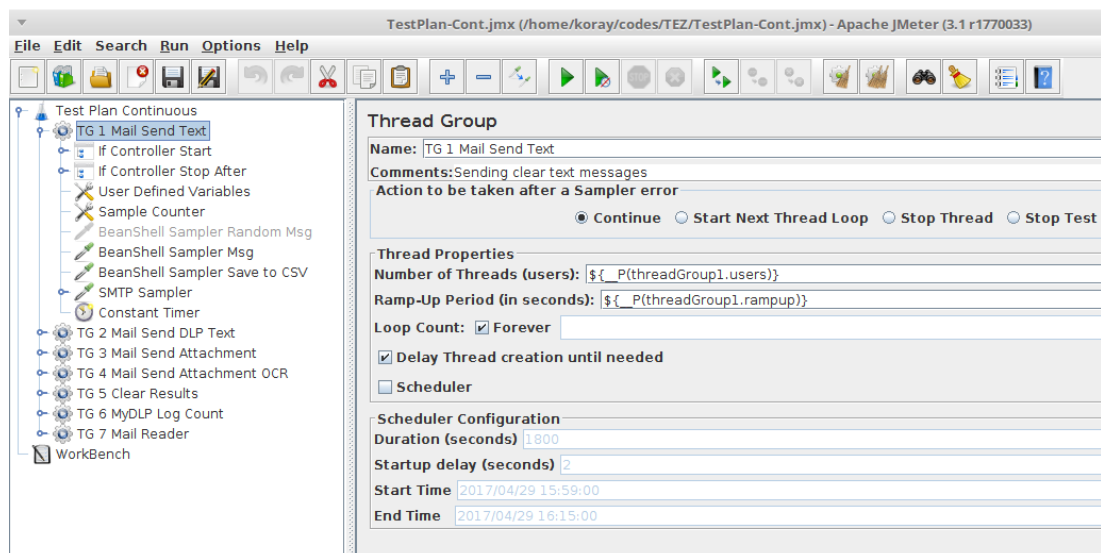


Figure 12 Test plan

Apache JMeter tests can be started from the command line as shown below:

```
jmeter -n -t TestPlan-Cont.jmx -p TestPlan_TG1.properties -Jsmtp.server=$1 -
Jsmtp.port=$2
```

4.4.2 Test scenarios:

Test scenarios are seen as thread groups in Fig. 12. Test scenarios are sequential combination of thread groups as shown below:

- Test Scenario 1: Send Text Message without attachments, Mail Read and Clear
 - TG1 + TG7 + TG5
- Test Scenario 2: Send Sensitive Text Message without attachments, Mail Read and Clear
 - TG2 + TG7 + TG5
- Test Scenario 3: Send Message with Sensitive Attachment, Mail Read and Clear
 - TG3 + TG7 + TG5
- Test Scenario 4: Send Message with Attachment OCR, Mail Read and Clear
 - TG4 + TG7 + TG5

Also the test types using thread groups is shown in Table 4.

	TG1	TG2	TG3	TG4	TG5	TG6	TG7
Email Relay without Filter	x	x	x		x		x
Email Relay with Filter	x	x	x	x	x		x
MyDLP with(out) Filter Test	x	x	x		x	x	x

Table 4. Test Thread Groups Matrix

4.4.3 Thread groups (TG)

There are seven thread groups in the test plan which are for sending (mails without attachments/Sensitive Text Message/Message with Sensitive Attachment), reading and clearing results and also a thread group for MyDLP as seen in Fig. 12. Threads are selected with the property files given to a bash script. Thread properties such as the number of users and ramp-up period can be changed within the thread property file. Also one thread group is running at a time.

Each test carried out with different threads (users) which are t2, t4, t6 and t8. For example, for a test of two threads (t2), there are 10 messages expected in the mailbox, for a test of 4 threads (t4) and there are 20 messages expected, etc. Test logic is implemented like injecting a paint droplet to a water pipe in one end and inspecting the droplet in the other end of water pipe. The tests which have sensitive messages are tagged with a "_DLP_" string so that one can expect to see the leakage in other end i.e. server.

Details of thread groups used in the test plan are summarised as follows:

- TG1 Mail Send Text (used for sending messages without attachments) and TG2 Mail Send DLP Text (used for sending messages with sensitive information)
 - If Control Start (Start if TG1/TG2 selected in property file)
 - If Control Stop After (If the sample counter is greater than 120, then stop)
 - User Defined Variables (the nonsensitive and sensitive text messages)
 - Sample Counter (Counter component)
 - BeanShell Sampler Msg (a java like script to select a text from the given user defined variables text value according to counter)
 - BeanShell Sampler Save to CSV (a java like script to append the selected text to a csv file for sharing information between threads)
 - SMTP Sampler (create a message with selected text, timestamp in subject and send to mail server)
 - Constant Timer (thread delay function)

- TG3 Mail Send Attachment (used for sending messages with sensitive attachments without OCR) and TG4 Mail Send Attachment OCR (attachment with OCR)
 - If Control Start (Check if TG3/TG4 selected in property file)
 - If Control Stop After (If the sample counter is greater than 120, then stop)
 - Sample Counter (Counter component)
 - BeanShell Sampler Attachment (a java like script to select a file from the given TEST and TEST_OCR folder)
 - BeanShell Sampler Save to CSV (a java like script to append the selected attachment file name to a csv file for sharing information between threads)
 - Constant Timer (thread delay function)
 - SMTP Sampler (create a message with the selected attachment, timestamp in subject and send to mail server)
 - Constant Timer (thread delay function)
- TG5 Clear Results
 - If Control Start (Check if TG5 selected in property file)
 - Mail Reader Sampler Delete (deletes the messages for the given mail server and account)
 - BeanShell Sampler CSV Reset (delete the csv files)
- TG6 MyDLP Log Count
 - If Control Start (Check if TG6 selected in property file)
 - HTTP Request Defaults (common parameters for sending multiple requests to the web server)
 - HTTP Post Login (creates a HTTP request to login with username and password to the given web page)
 - HTTP Cookie Manager (not to login before every HTTP request)
 - HTTP Request Get Logs (an HTTP request to get the json log web page)
 - Total Log Records JSON Extractor (count the total log items in the json)
 - BeanShell Sampler Save to CSV (save the results to a csv file)
- TG7 Mail Reader (used for reading the messages)
 - If Control (Check if TG4 selected in property file)

- While Controller (Wait for 120 messages come to inbox)
- Mail Reader Sampler (reads the given count of mails for the given mail server and account)
 - BeanShell PostProcessor MAILR (for each of the message calculate the difference between the timestamp in subject and arrival time in the mail server)
 - BeanShell Sampler MailCount (calculate the received mail count)
- BeanShell DLP Mail Count (count the X-MailRelay-DLP in the header)
- BeanShell Mail DLP Count Assertion (assert if the sensitive mail count differs from the expected)
- Assertion Results (list the assertions)
- BeanShell Sampler CSV (append the results to a csv file)

We used 120 message samples for each test scenario. Number of threads are represented with t#, namely t2 is used for two threads, t4 is used for 4 threads, etc. Threads in JMeter is equivalent to a user. Number of threads parameter is set from the property file which can be shown in Fig. 12. Ramp-up period in Fig. 12 is the delay between starting each user or thread. We selected the ramp-up time as 1 second. So having 4 threads and 1 second ramp-up time there will be 4 user per second. Loop count is set to forever and after 120 sample count, the SMTP sampler threads (TG1, TG2, TG3 or TG4) are stopped. Then the mail reader thread group (TG7) starts for reading the messages. The messages' headers are processed after reading with Mail Reader Sampler in the TG7 thread group. The delay time from sending to the receiving mail is calculated for each sample in TG7 thread group. The sender thread group's timestamp and the received timestamp at the mail server are subtracted while processing the mail header. In addition to mail delay time, the expected leakage header is searched from the header of the mail for the email relay case. Since email relay filter script adds a "X-MailRelayDLP: 1" header to the message before sending if the mail is sensitive.

For the MyDLP case, the leakage information can be stored in the MyDLP web log page. Log records can be seen in the /logs web page and all log records can be seen in JSON format in /api/logs page. Before the mail reader sampler starts, the TG6 MyDLP Log Count starts so that the sensitive data log count retrieved. After sending the mails, new sensitive log count is retrieved. Leak count is found from the difference before sending and after sending of mail. In the TG7 thread group, the known leakage information and the expected leakage information is compared for assertion purposes. If the expected leakage count is the same as calculated one, then there is no assertion.

4.5 Test Results

The TG7 Mail Reader thread group stores the results in a file for each test. The results are the mail delay time, assert count, mail send log and the mail receive log. The subsequent charts show the mail delay times (milliseconds) for each threads.

4.5.1 Email Relay No Filter Test

In this no filter test, test scenario 1, test scenario 2 and test scenario 3 are tested. The tests are executed for the EmailRelay server without leakage detection script to see the server performance for a baseline. The Fig. 13-15 shows message vs delay time of message chart for sending small sized text messages for different user loads. As the user load increase, message delay increases as expected. Also there is no pre-process (filter script) running for a message. The patterns are almost the same. Fig. 15 shows on average the delay times are equal because of the nearly same sized messages.

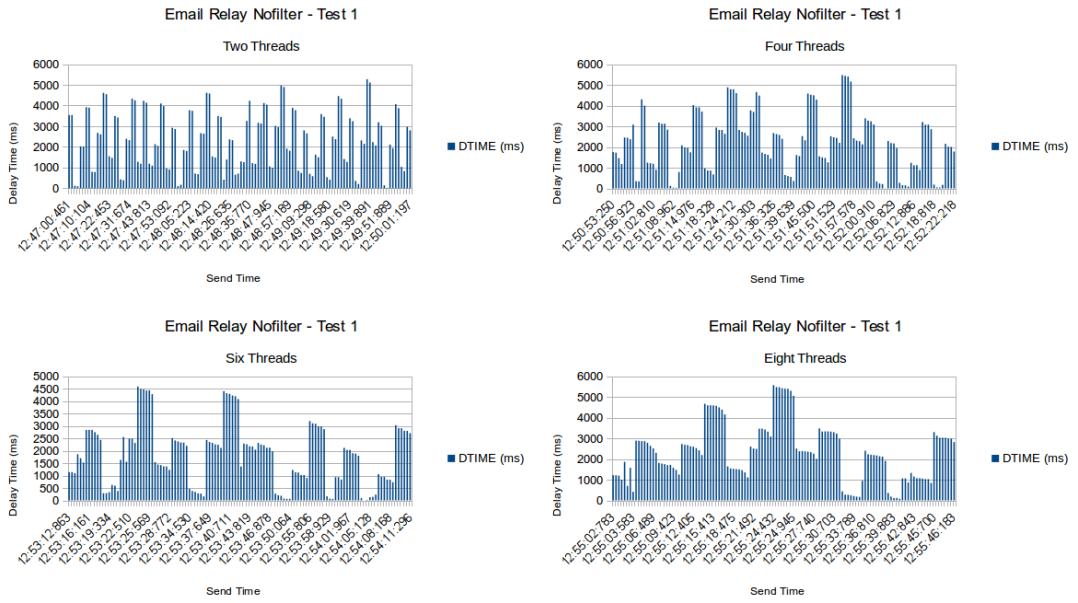


Figure 13 Email Relay No Filter Test 1

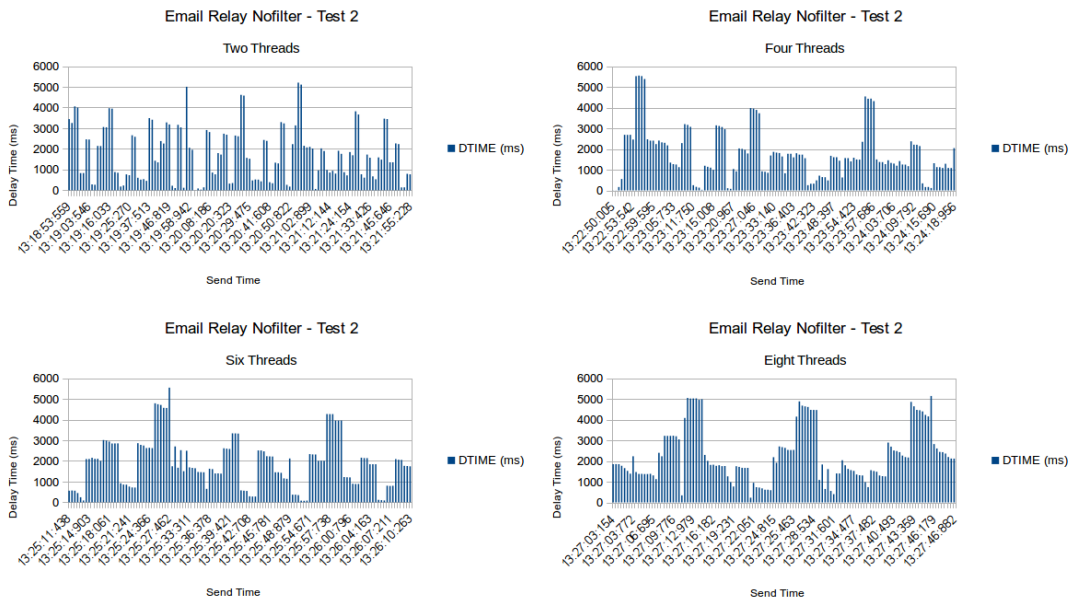


Figure 14 Email Relay No Filter Test 2

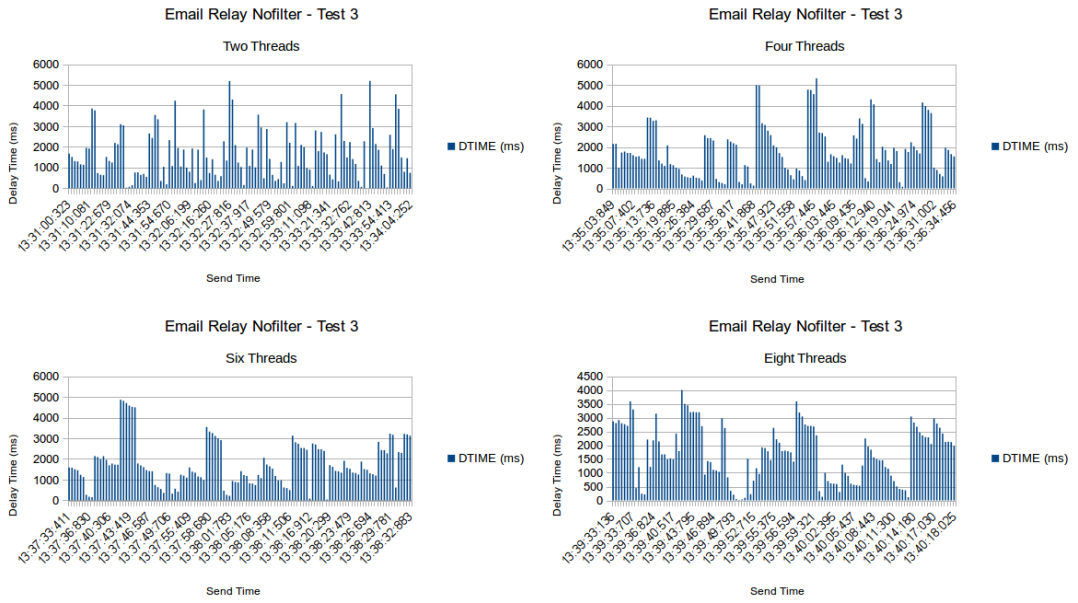


Figure 15 Email Relay No Filter Test 3

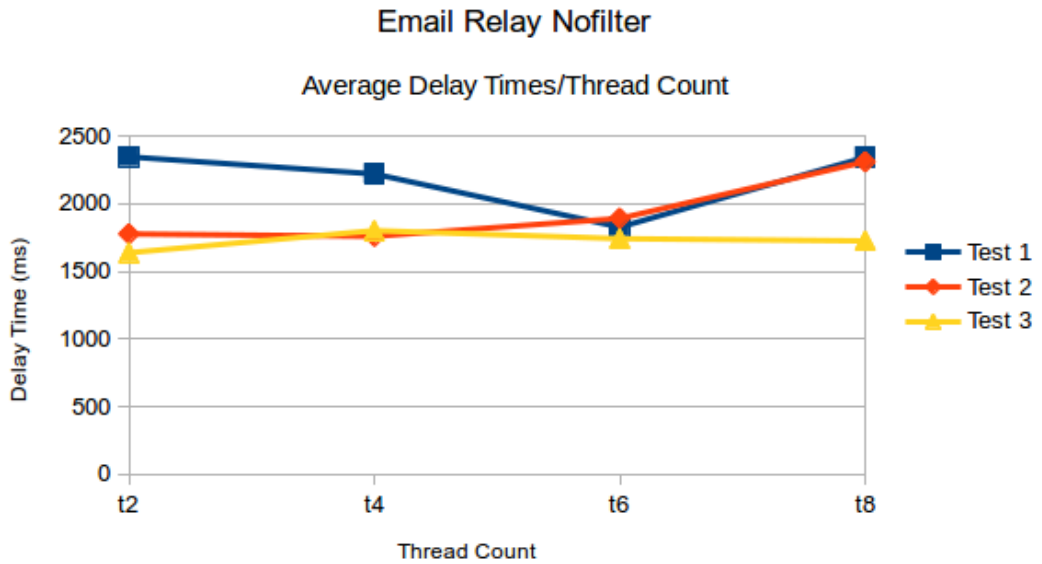


Figure 16 Email Relay No Filter Average Delay

4.5.2 Email Relay Local Parse Test

In this test all of the test scenarios are tested. In Fig. 17-20 test results of messages for email relay local parse architecture can be seen. In Fig. 17 there is no sensitive message in injected but filter script is running also. In comparison to Fig. 13, Fig. 17 has a little higher delay times because of the filter script. In test 2, the messages have some sensitive information like TC ID numbers or SGK numbers. As seen Fig.17 and Fig.18 there's no big difference in terms of delay time for tests 1 and 2. Likewise this similarity can be seen in Fig. 21. On the other hand the test 3 which is sending attachments test have some dramatic impact on resources so that the delay increases as shown in Fig. 19. Moreover if the attachment has as embedded image as in test 4, the delay times are getting longer as shown in Fig 20.

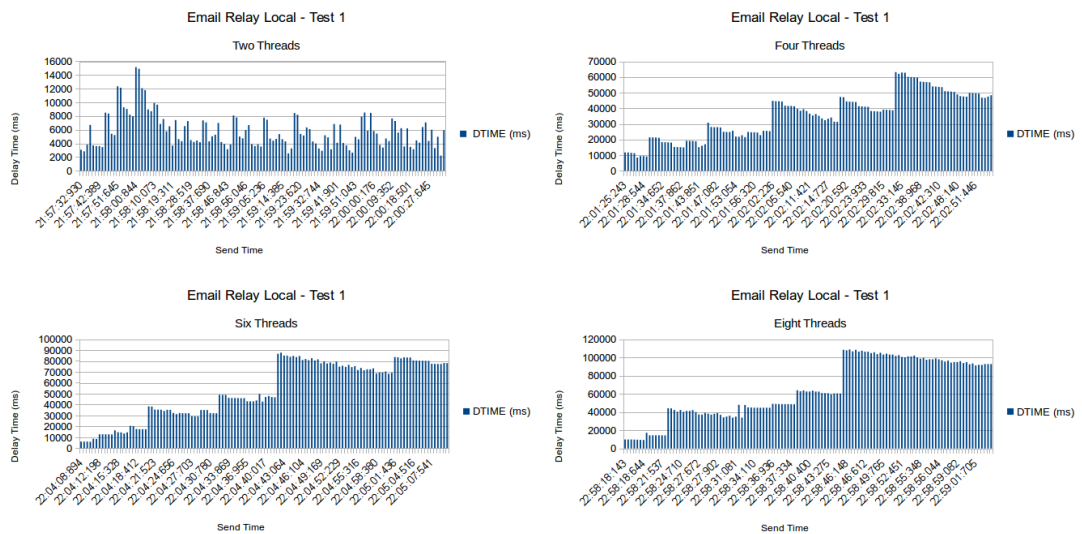


Figure 17 Email Relay Local Parse Test 1

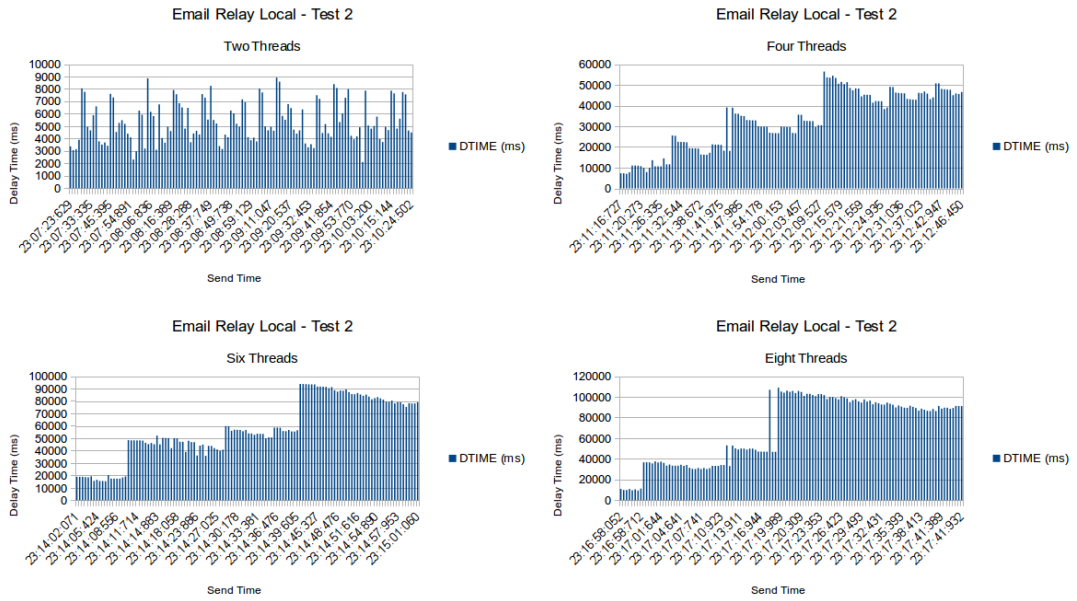


Figure 18 Email Relay Local Parse Test 2

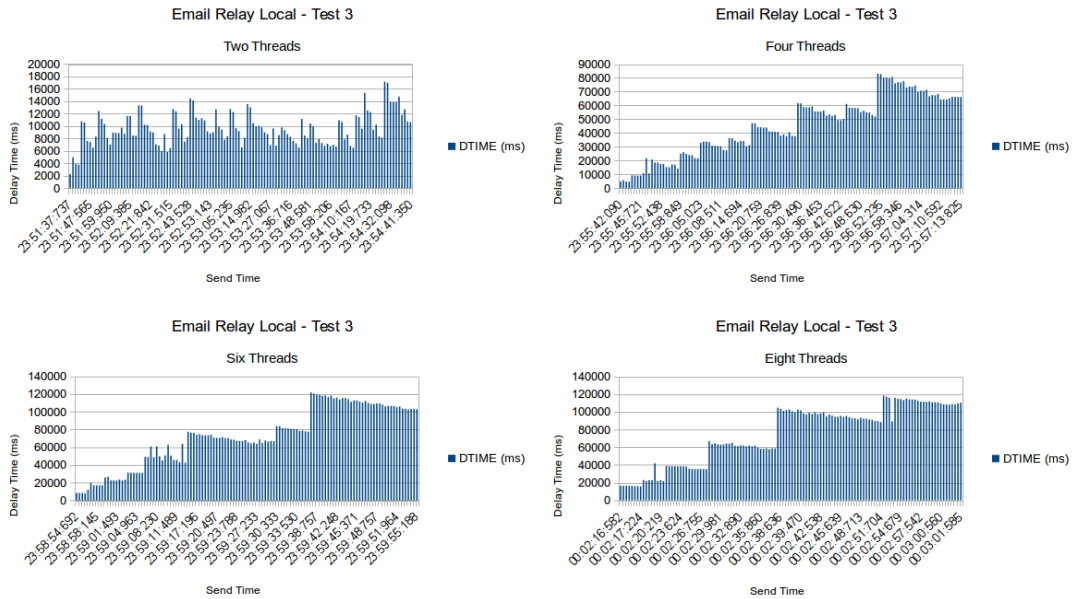


Figure 19 Email Relay Local Parse Test 3

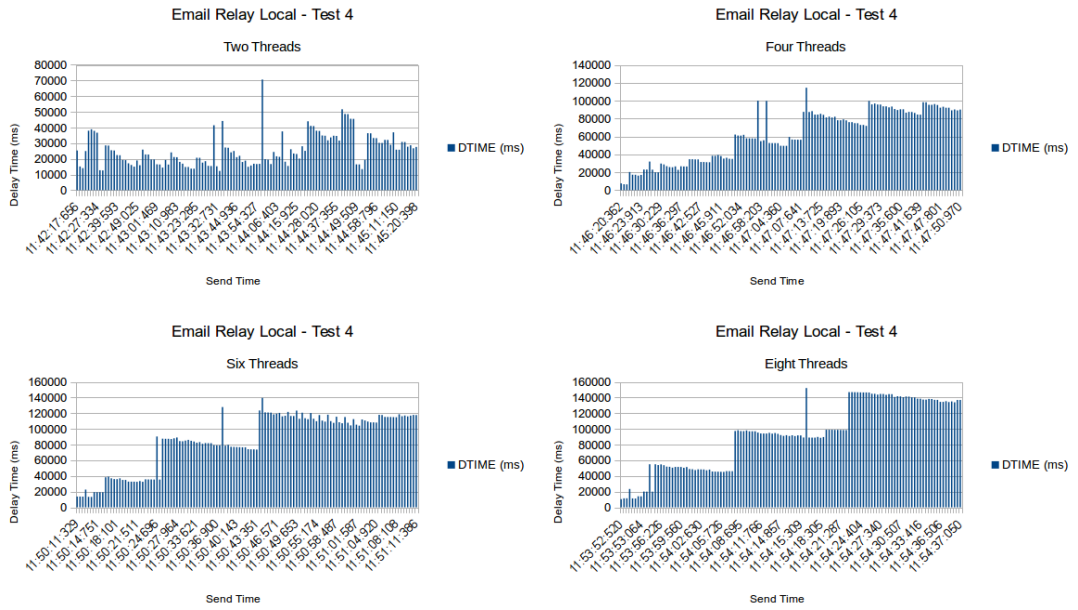


Figure 20 Email Relay Local OCR

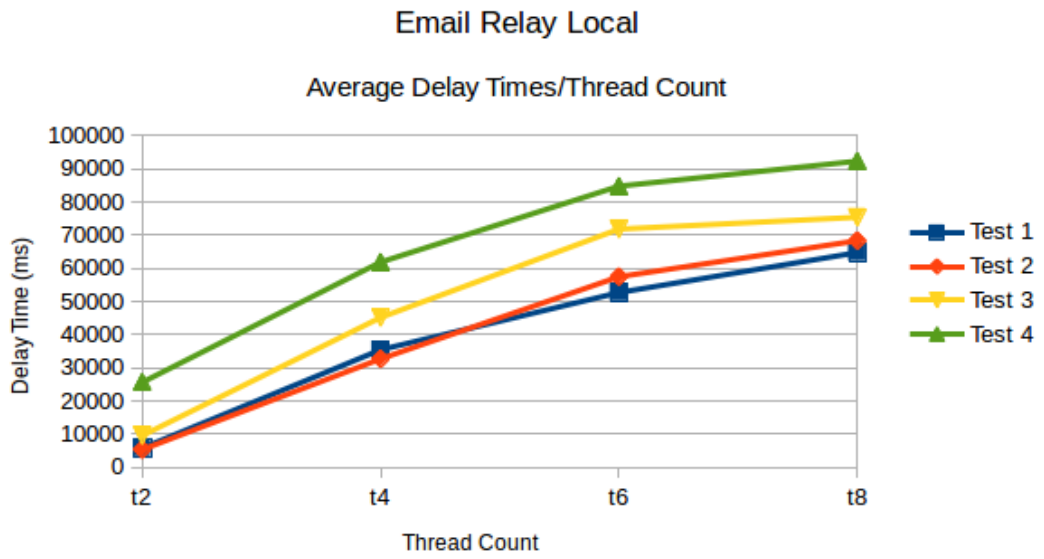


Figure 21 Email Relay Local Average Delay

4.5.3 Email Relay Remote Parse Test

In this test all of the test scenarios are tested. In Fig. 22-26 the email relay remote parse architecture test results can be seen. The e-mail relay outgoing server communicates with a remote parser server to forward a message or not.

The average message delay time difference between local and remote parse increases linearly with thread number for test 1, test 2 and test 3. For test 1, test 2 and test 3 the thread number does not have a big impact on the remote parse server of email relay server. Test 3's delay times are larger than test 1 and test 2; because attachment files are processing. Also it is more efficient than local parse architecture as seen in Fig. 24 with comparison to Fig. 19 for t6 and t8. Moreover test scenario 4 is more time consuming because of the embedded images in the files. These files are also extracted with pdfimages [51] command and getting text (OCR) from them. The processing of images have a significant burden on the system as seen in Fig. 25 and Fig. 26. Since OCR is a CPU bound problem, the system's CPU have to be increased or another OCR server have to added.

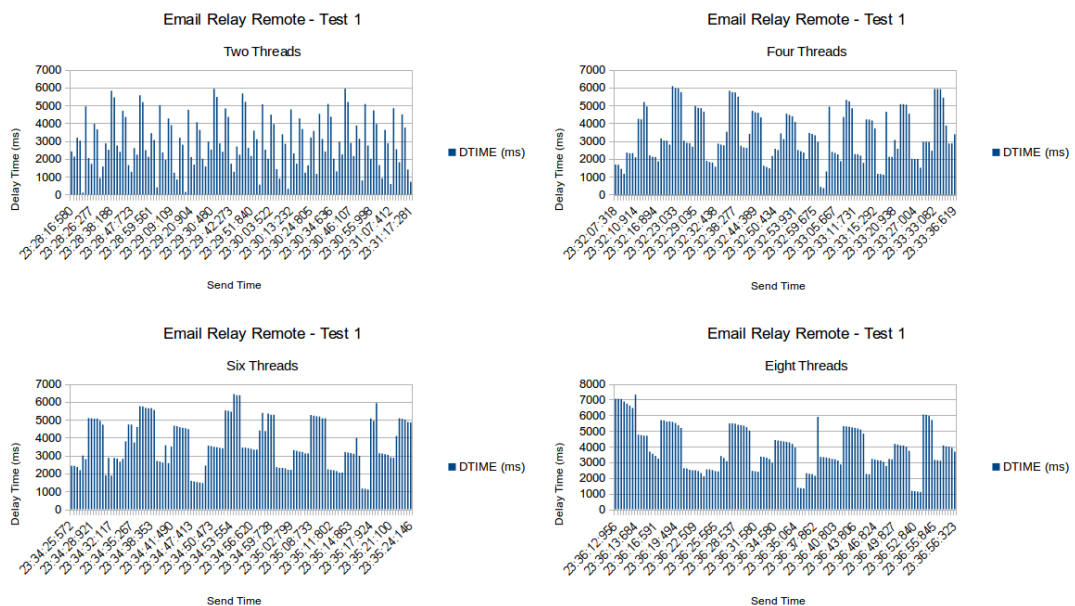


Figure 22 Email Relay Remote Parse Test 1

Also in Fig.24-25 some high valued spikes are shown up. This is because of the error 452 message (message processing failed: read error disconnected) thrown by the e-mail relay server. After this error the message file extension is changed to “.bad”. E-mail relay server provides the notification and resending these kind of messages to the recipient with bash scripts. These scripts are run by the cron job scheduler. So these high spike delays from the cron job.

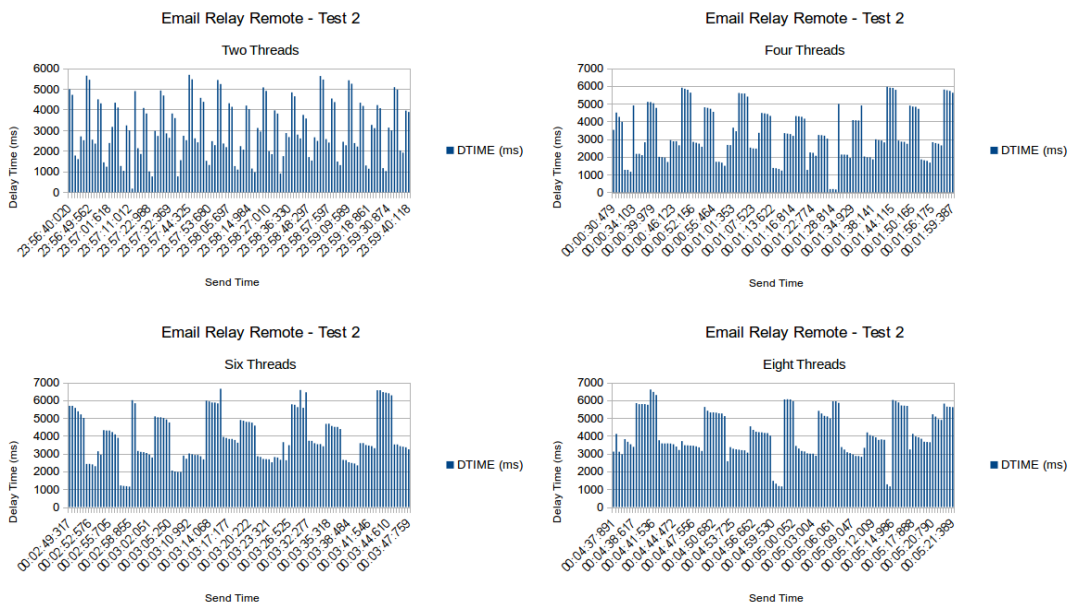
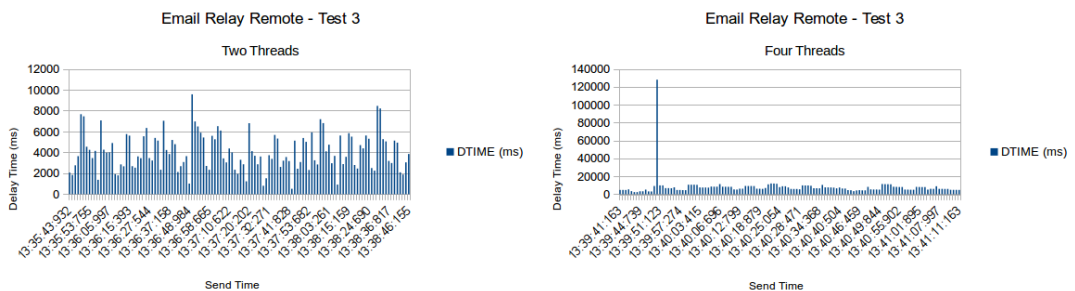


Figure 23 Email Relay Remote Parse Test 2



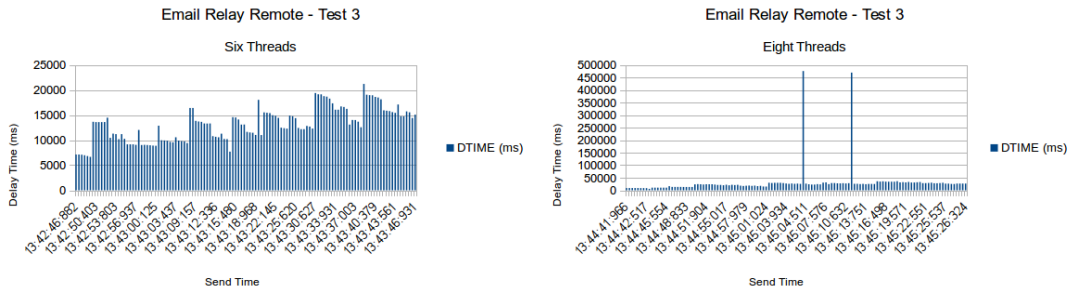


Figure 24 Email Relay Remote Parse Test 3

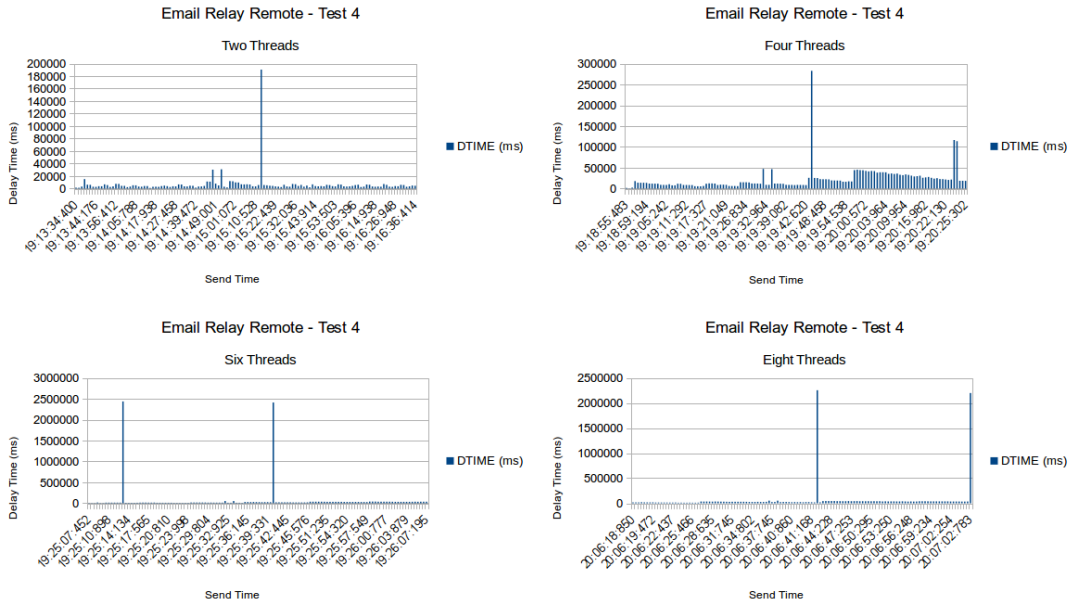


Figure 25 Email Relay Remote Test 4 OCR

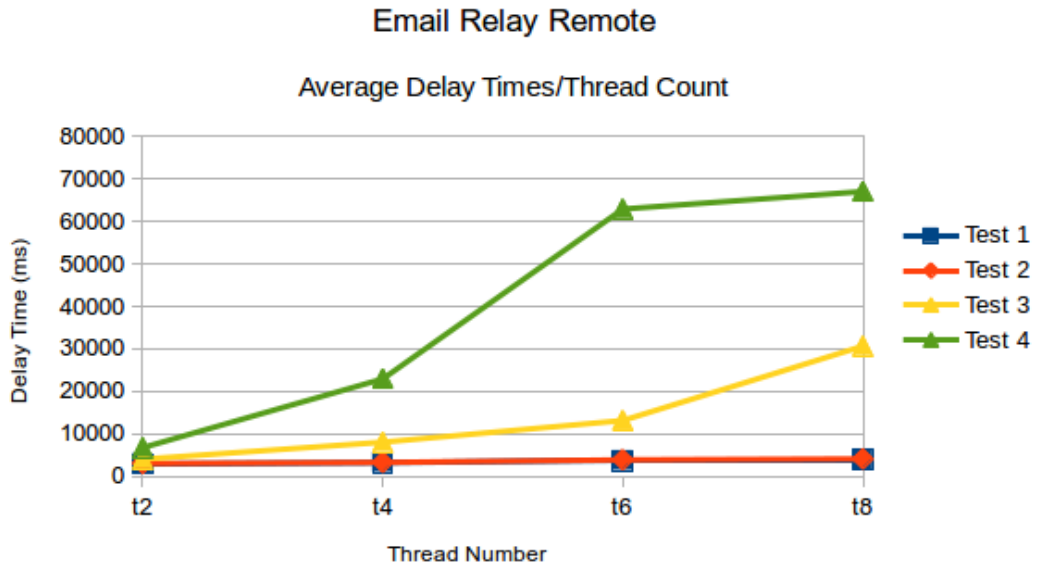
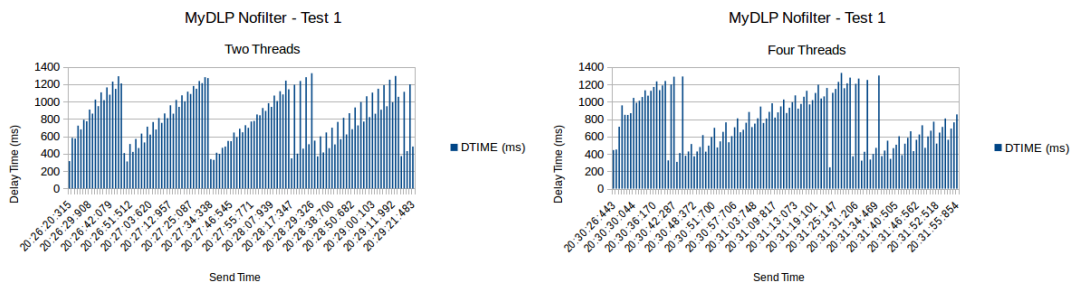


Figure 26 Email Relay Remote Parse Average Delay

4.5.4 MyDLP No Filter Test

MyDLP's SMTP server without content filtering is also tested for comparison. It is a Postfix server actually. In order to disable the filtering, the Postfix main.cf is changed.

In this test all of the tests except the Test 4 OCR are tested. In this test it is the delay times are expected to be lower than the MyDLP with content filter. The test results are shown as Fig 27-30.



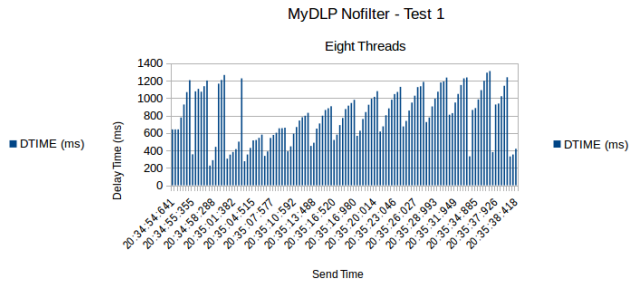
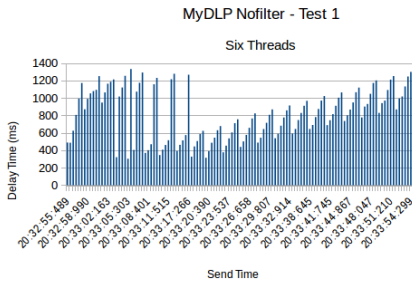


Figure 27 MyDLP No Filter Test 1

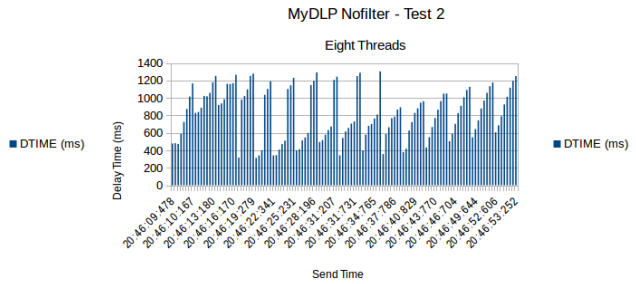
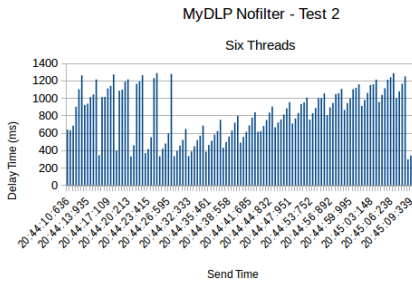
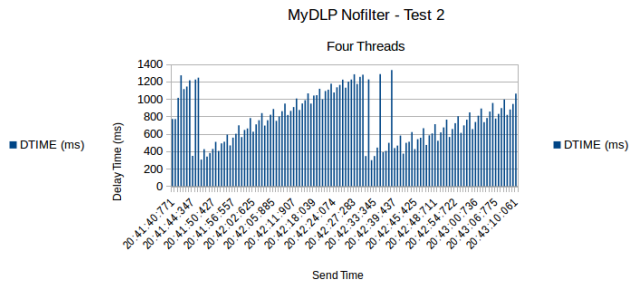
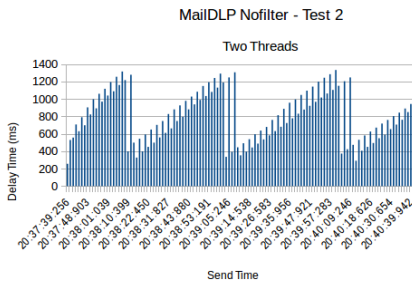
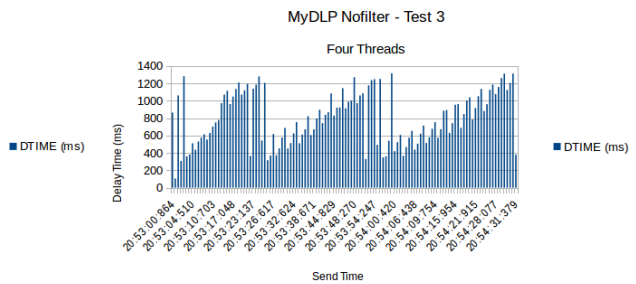
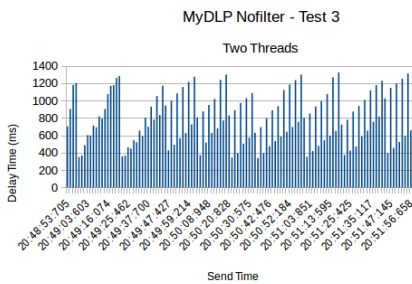


Figure 28 MyDLP No Filter Test 2



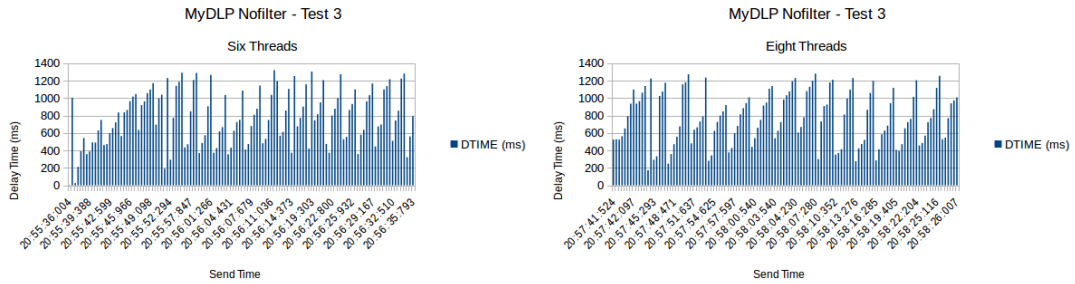


Figure 29 MyDLP No Filter Test 3

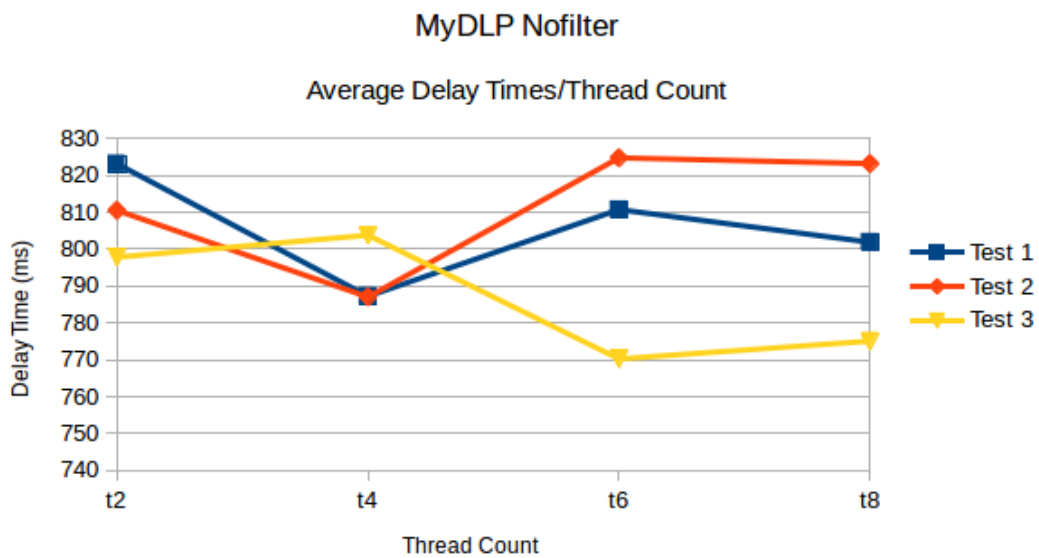


Figure 30 MyDLP No Filter Average Delay

4.5.5 MyDLP Test

MyDLP uses postfix e-mail system. Postfix content filter option is set to MyDLP server socket. This can be seen with running “postconf content_filter” command in the MyDLP appliance terminal which is “content_filter = dlp:[127.0.0.1]:10026”. When an e-mail is sent using the Postfix SMTP server, the message is first redirected to the MyDLP server (a server listens on 10026) for content inspection. If no sensitive content is discovered, the message is forwarded to the original recipient [7].

In this test all of the test scenarios are tested except the OCR test. Since MyDLP 3.2.0 does not support matching files with images. Also in this test only MyDLP server's email leakage feature is tested.

In Fig. 31-34 the MyDLP test results can be seen. The Postfix server communicates with the local MyDLP server. This architecture is similar to the e-mail relay local parse architecture. The results seen in Fig. 31-33 have better delay times than the e-mail relay server with parsing. In Fig. 27 initial values are bigger than the latter ones.

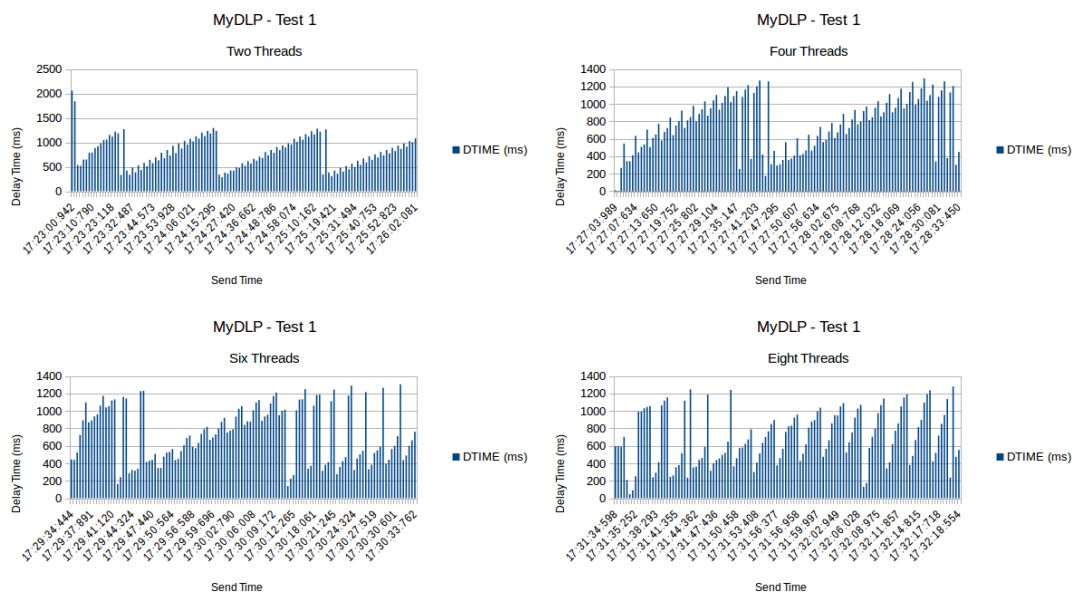
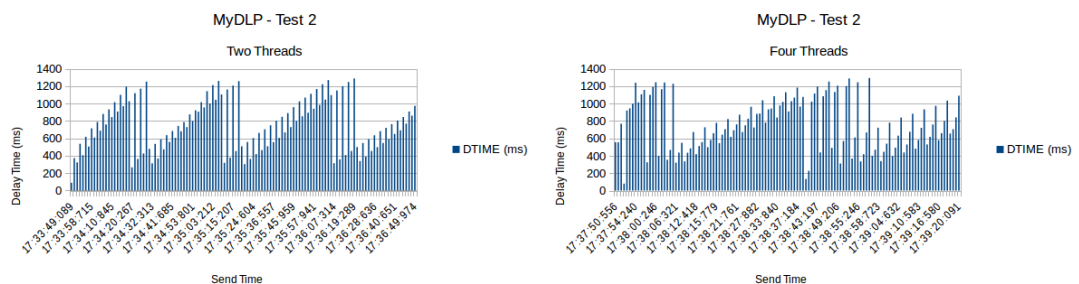


Figure 31 MyDLP Test 1



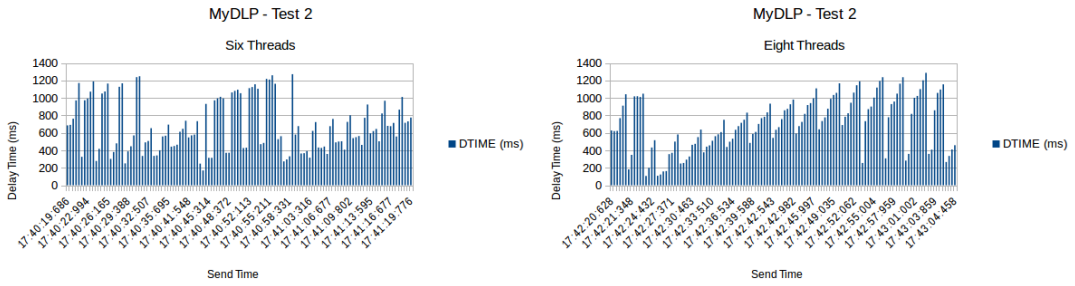


Figure 32 MyDLP Test 2

In Fig. 34 the average delay times per tests can be shown. There is at most one second difference in Fig. 31 and Fig. 32 which can be for the logging the leakage data in Fig. 32. The MyDLP server average delay times is much lower than the email relay parse architectures. The detailed comparisons are also shown in Fig. 35-38.

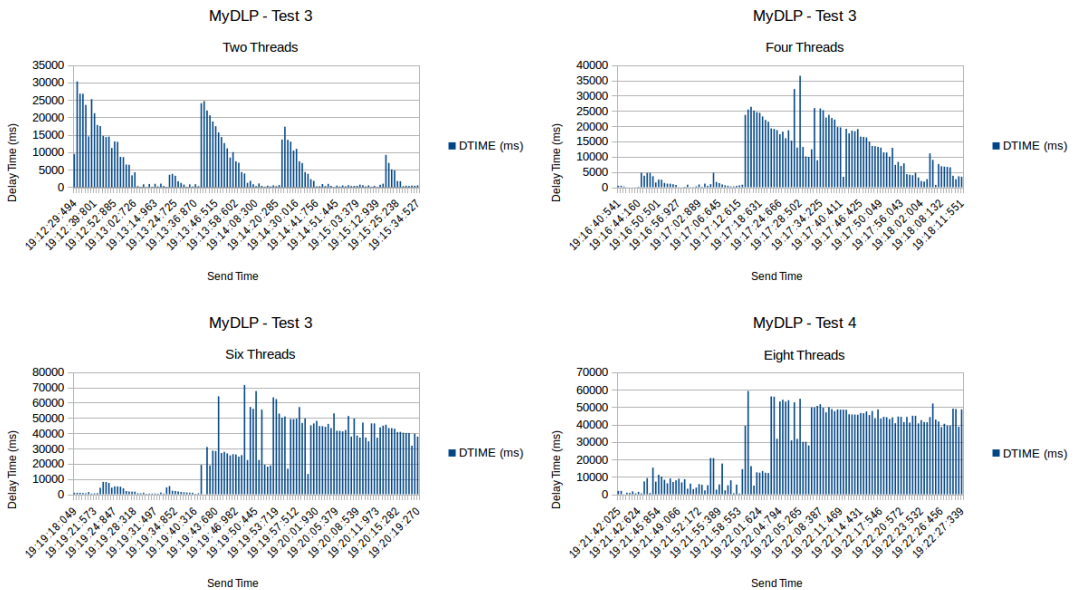


Figure 33 MyDLP Test 3

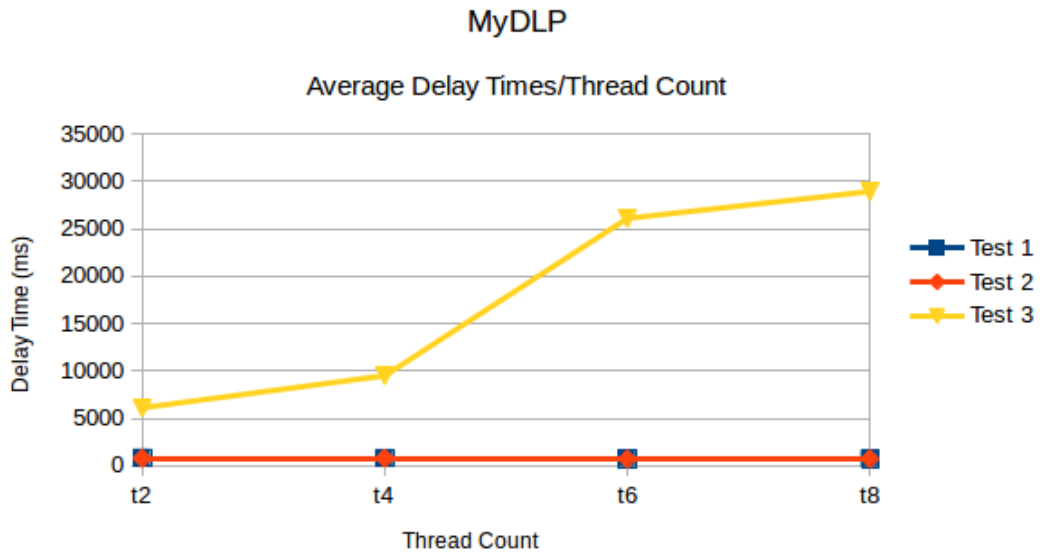
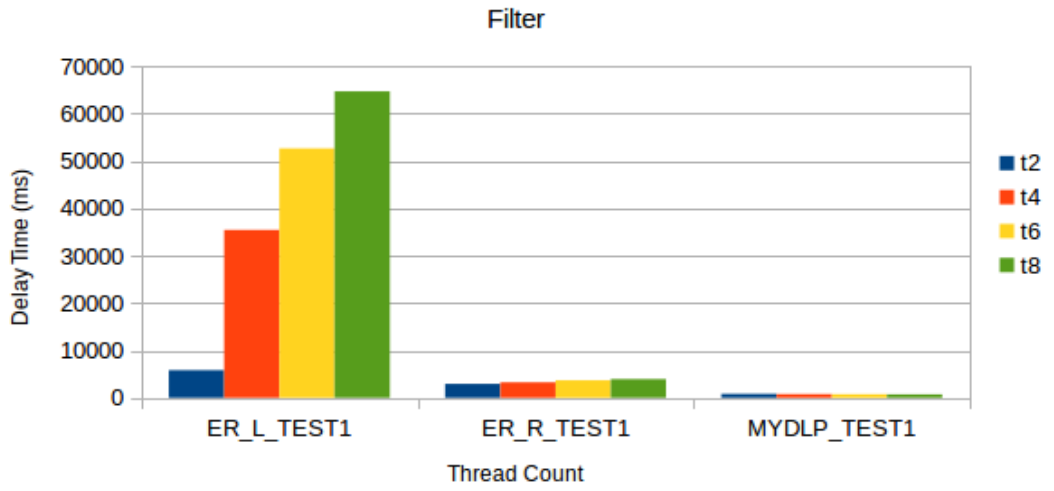


Figure 34 MyDLP Average Delay

4.6 Test Evaluation

Email relay local and remote parse architectures and MyDLP email DLP system were tested with different loads. For t2, t4, t6 and t8 threads and for different test scenarios, the average delay time for remote parse architecture is lower than local parse architecture as seen in Fig 35-38. This is an expected result; because using a network pre-processor compared to a running program is that the Email relay server is not blocked while the messages are being pre-processed [3]. On the other hand without filter pre-preprocessing the email relay server average delay times are a little higher than MyDLP server as seen in Fig. 35-38.

Test 1 Average Delay Comparison



Test 1 Average Delay Comparison

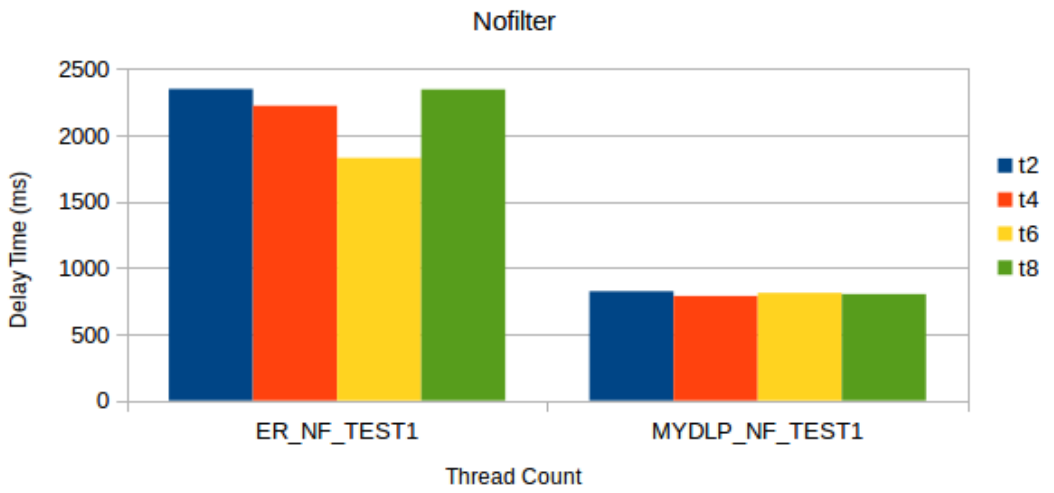


Figure 35 Test Scenario 1/2 Results

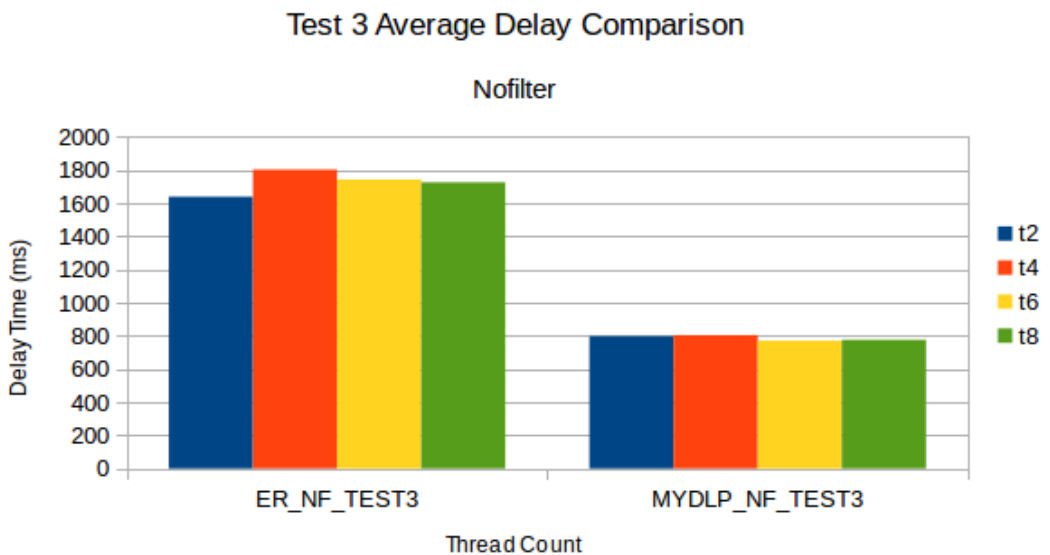
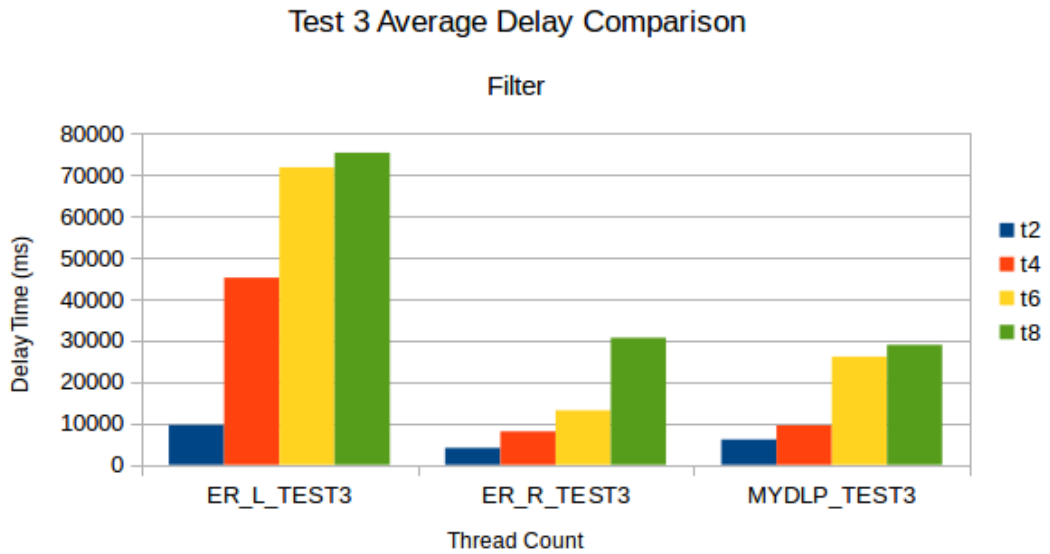


Figure 36 Test Scenario 3 Results

The test scenario 3 is shown in Fig. 36 and it can be seen from the figure that local parsing delays are much higher than the remote parsing and MyDLP after two threads/users.

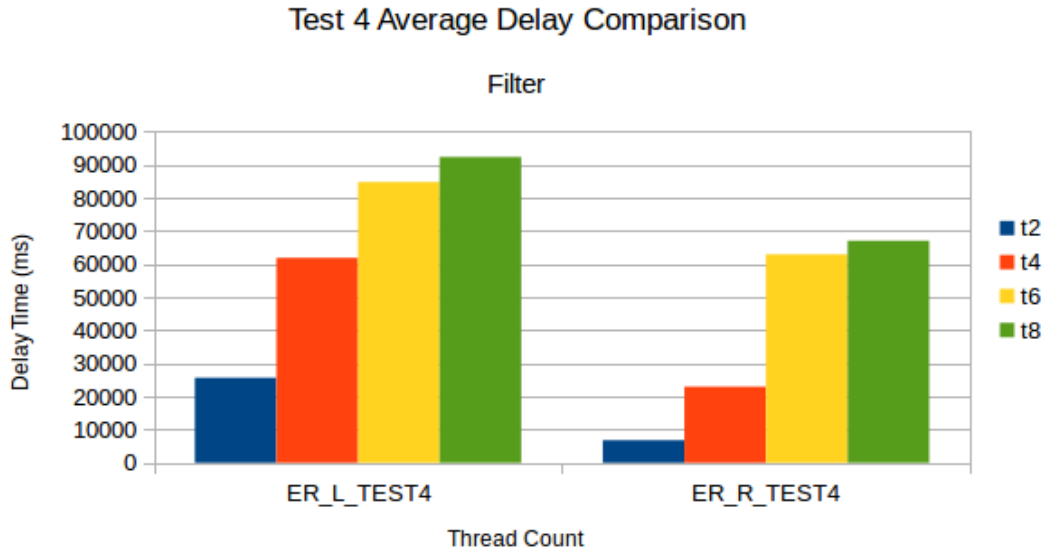


Figure 37 Test Scenario 4 Results

The test scenario 4 is the same third test repeated with most of the same PDF type documents and extra OCR PDF documents. These OCR documents have images embedded. In this test MyDLP did not detect the TC ID pattern. On the other hand the e-mail relay filter script found the TC ID pattern successfully. Also it can be seen from Fig. 37 that for different loads the remote parsing architecture is better than local parsing architecture.

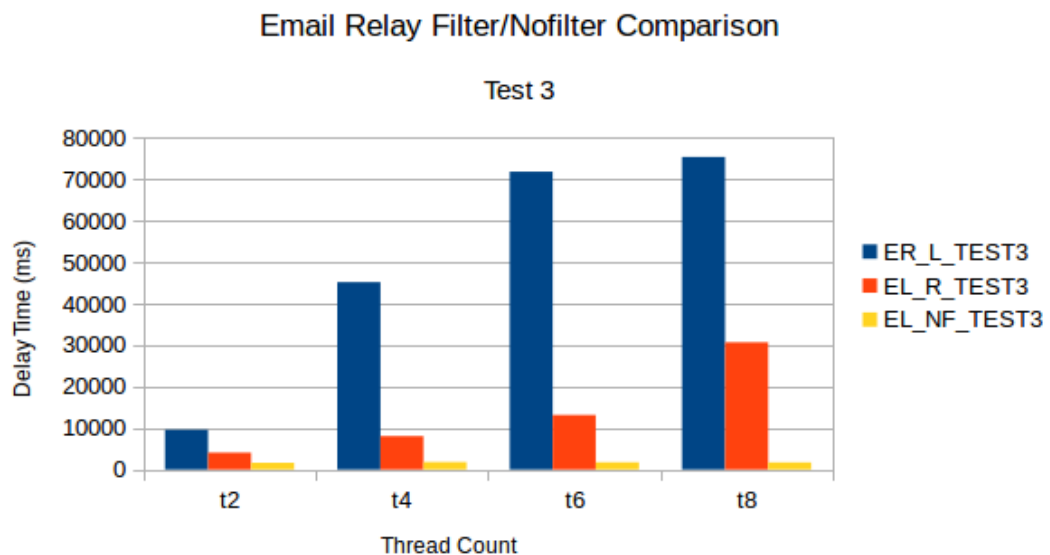
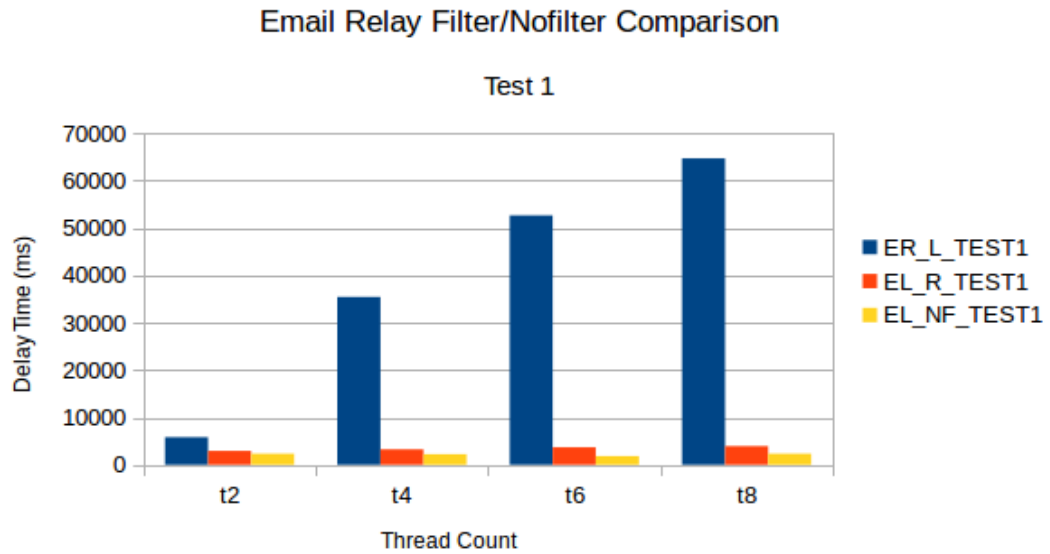


Figure 38 Email Relay Filter/No Filter Comparison

In Fig. 38 above the e-mail relay server with and without filter script running is compared. The test 2 is not added to the figure; because it is almost the same as test 1. The local parsing architecture has high delay values as expected because of long-running script and blocking of e-mail server. Since the number of e-mail relay instances are increased, it has also high values of delay as shown in Fig. 38. Also the no filter delay value for test 3 and t2 threads is 1/2.5 times of remote parsing value which is a good result.

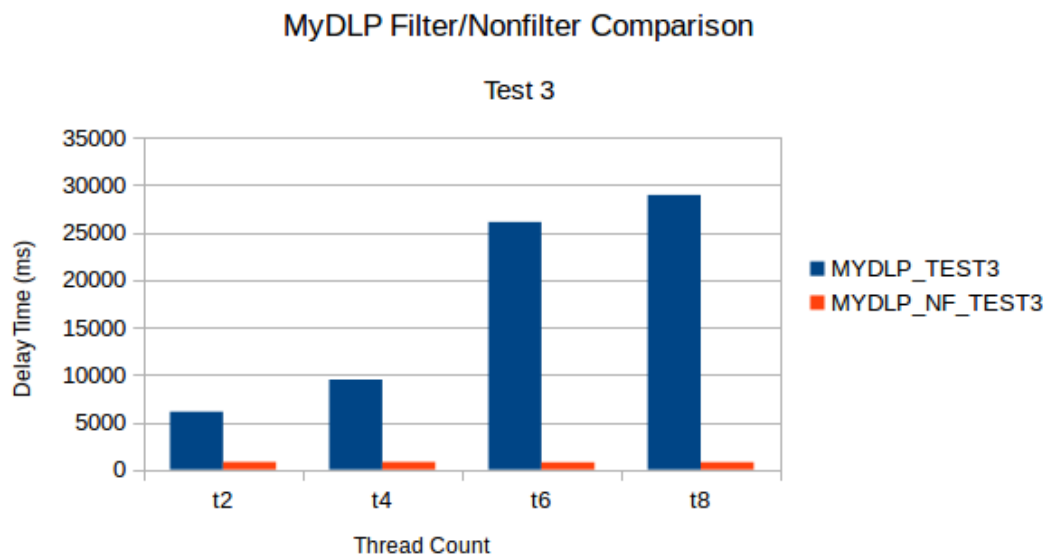
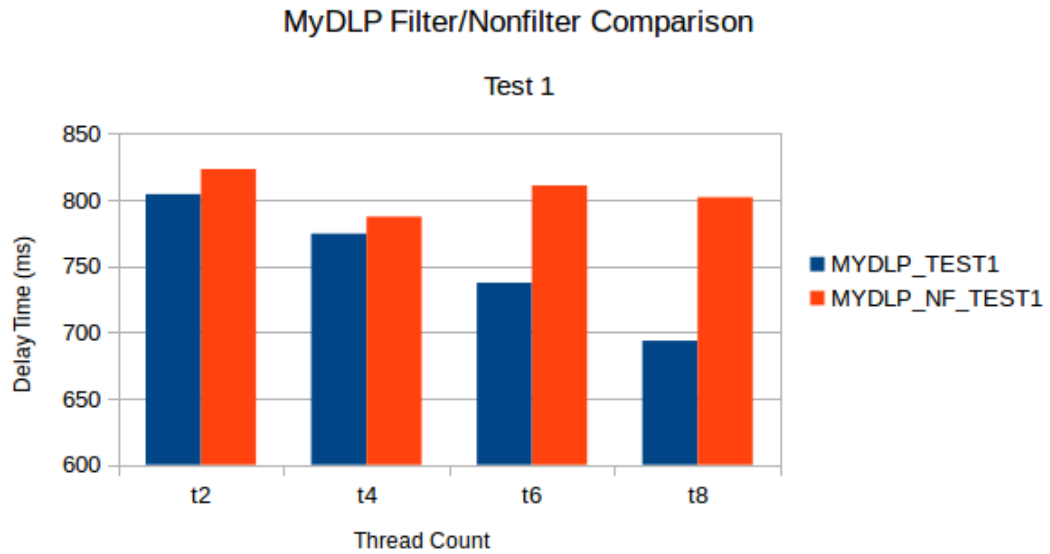


Figure 39 MyDLP Filter/No Filter Comparison

In Fig. 39 the MyDLP with and without filter, i.e. Postfix mail server is compared. The test 2 is not added to the figure, since it is same as test 1 again. It is seen from Fig. 39. that only the messages having attachments (test 3) are so different each other. For test 3 and t2 threads, while the MyDLP server has 6102 ms average delay, the MyDLP without filter (postfix with content_filer="" set) has 798 ms which is 8 times faster as expected. Also after 4 threads (t4), the MyDLP without filter is 34 times faster than MyDLP with filter.

CHAPTER 5

CONCLUSION AND FUTURE WORK

5.1 Conclusion

In this work, e-mail data leakage vector is investigated. For that purpose an SMTP proxy server is searched and an open source solution written by Graeme Walker found and used. Then to parse the incoming e-mails, a parser script written. Documents and text having 11-digit TC ID numbers are tested with different loads. In order to reduce the false positives, a TC ID check function is used. On the other hand, another check is required in order to reduce false positives. The context of the message have to be also analysed in order to reduce false positives. Also open source solution MyDLP's SMTP DLP module is compared with EmailRelay.

According to the test results, it is better to use remote parsing architecture for low delays. All of the messages sent to EmailRelay server successfully processed and forwarded. Also all of the messages sent to MyDLP server successfully forwarded too. On the other hand in the time of testing, MyDLP did not look at the embedded images in the documents. So it missed some of the scanned documents with sensitive information.

The implementation and the design of the proposed solution is so simple that one can add any control to the logic. For example html part of the e-mail did not processed. One can also look at these to find malicious links in addition to find data leakage. Moreover with the chain design of the e-mail relay system, one can also add different

control layers like TCP/IP stack. For example in one layer rule based controls are done and in another layer behaviour based controls can be done. Since the solution we used can alter the e-mail file before sending to the destination. Also our script has been written not only for e-mail relay server but also in any smtp proxy based servers.

Our approach can also be deployed in a cloud so that cloud advantages can be used. We presented a scalable open source free email DLP solution which have as much as features as the above solutions. Since the library and projects used are open source, it can also be extended.

5.2 Future Work

In this study, only the content analysis of the e-mail messages have been conducted. There are also false positive results for the regular expression rules. So in order to avoid these, one can also use naive bayesian or decision tree learning techniques for classifying the sensitivity of the e-mail message. In the future work, the data set must be prepared for the different sector corpora, i.e. finance confidential, health information, person names, person addresses, etc. which have to be classified first by hand. For this purpose "Hizmete Özel" and "Tasnif Dışı" keywords searched through Google and downloaded from Google. A data set is prepared with batch converting these to textual data. Although a not so clean data set prepared, the classification accuracy is not good enough (%68) to use. If it was good enough, a second chain of e-mail relay server could be added which the classifier script run. If there is a pattern from the first layer, then it will look at the second layer also so that the accuracy of leakage finding can be improved contextually.

REFERENCES

- [1] Walker, G. (2008, April 15). E-Mail Relay User Guide. Retrieved May 17, 2017, from <http://emailrelay.sourceforge.net/userguide.html>
- [2] Mattmann, C. (2017, April 05). GitHub Repository, Chrismattmann/tika-python. Retrieved May 18, 2017, from <https://github.com/chrismattmann/tika-python>
- [3] Walker, G. (2008, April 18). Email Relay Reference. Retrieved May 18, 2017, from <http://emailrelay.sourceforge.net/reference.html>
- [4] Tahboub, R., & Saleh, Y. (2014, January). Data leakage/loss prevention systems (DLP). In *Computer Applications and Information Systems (WCCAIS), 2014 World Congress on* (pp. 1-6). IEEE.
- [5] Koutsourelis, D., & Katsikas, S. K. (2014, October). Designing and developing a free Data Loss Prevention system. In *Proceedings of the 18th Panhellenic Conference on Informatics* (pp. 1-5). ACM
- [6] Gordon, P. (2007). Data Leakage-Threats and Mitigation. *InfoSec Reading Room*.
- [7] Torsteinbo, T. (2012). Data Loss Prevention Systems and Their Weaknesses. *University of Agder*.
- [8] Polatcan, O. (2011). E-mail behavior profiling: Based on attachment and language text.
- [9] Elmrabit, N., Yang, S. H., & Yang, L. (2015, September). Insider threats in information security categories and approaches. In *Automation and Computing (ICAC), 2015 21st International Conference on* (pp. 1-6). IEEE.
- [10] William, S. (2008). *Computer Security: Principles And Practice*. Pearson Education India.
- [11] External threat. (n.d.). Retrieved May 18, 2017, from http://itlaw.wikia.com/wiki/External_threat

- [12] Threat (computer). (2017, May 16). Retrieved May 18, 2017, from [https://en.wikipedia.org/wiki/Threat_\(computer\)](https://en.wikipedia.org/wiki/Threat_(computer))
- [13] John, R. V. (2009). Computer and information security handbook. *Morgan Kaufmann, May, 29*, 739.
- [14] Kissel, R. (2013). Glossary of key information security terms. *NIST Interagency Reports NIST IR, 7298(3)*.
- [15] User, A., & Planning, A. (1995). An Introduction to Computer Security: The NIST Handbook.
- [16] Whitman, M. E., & Mattord, H. J. (2011). *Principles of information security*. Cengage Learning.
- [17] Yang, Y., & Joachims, T. (2008). Text categorization. *Scholarpedia*, 3(5), 4242.
- [18] Data classification (data management). (2017, February 01). Retrieved May 18, 2017, from [https://en.wikipedia.org/wiki/Data_classification_\(data_management\)](https://en.wikipedia.org/wiki/Data_classification_(data_management))
- [19] Document classification. (2017, April 07). Retrieved May 18, 2017, from https://en.wikipedia.org/wiki/Document_classification
- [20] Statistical classification. (2017, May 15). Retrieved May 18, 2017, from https://en.wikipedia.org/wiki/Statistical_classification
- [21] Data Classification. (n.d.). Retrieved May 18, 2017, from <https://www.research.ibm.com/haifa/projects/imt/sdc/index.shtml>
- [22] Apache Tika. (2017, April 28). Retrieved May 18, 2017, from https://en.wikipedia.org/wiki/Apache_Tika
- [23] Gormley, C., & Tong, Z. (2015). Elasticsearch: The Definitive Guide. " O'Reilly Media, Inc."
- [24] Hart, M., Manadhata, P., & Johnson, R. (2011, July). Text classification for data loss prevention. In *International Symposium on Privacy Enhancing Technologies Symposium* (pp. 18-37). Springer Berlin Heidelberg.
- [25] Aoe, J. I. (1994). Computer algorithms: string pattern matching strategies (Vol. 55). John Wiley & Sons.
- [26] Friedl, J. E. (2006). Mastering regular expressions. " O'Reilly Media, Inc."

- [27] Shabtai, A., Elovici, Y., & Rokach, L. (2012). A survey of data leakage detection and prevention solutions. Springer Science & Business Media.
- [28] Trend Micro Data Protection Whitepaper. (n.d.). Retrieved May 18, 2017, from https://www.eiseverywhere.com/file_uploads/42b19bab0e9593837735ee5fcf0e63ea_Trend_Micro.pdf
- [29] Shapira, Y., Shapira, B., & Shabtai, A. (2013). Content-based data leakage detection using extended fingerprinting. *arXiv preprint arXiv:1302.2028*.
- [30] Cisco IronPort E-mail Data Loss Prevention. (n.d.). Retrieved May 18, 2017, from http://www.cisco.com/c/dam/en/us/products/collateral/security/email-security-appliance/Cisco_IronPort_Email_Data_Loss_Prevention_QA.pdf
- [31] Raman, P., Kayacık, H. G., & Somayaji, A. (2011, June). Understanding data leak prevention. In *6th Annual Symposium on Information Assurance (ASIA'11)* (p. 27).
- [32] Alneyadi, S., Sithirasanen, E., & Muthukkumarasamy, V. (2016). A survey on data leakage prevention systems. *Journal of Network and Computer Applications*, 62, 137-152.
- [33] Kleene, S. C. (1951). *Representation of events in nerve nets and finite automata* (No. RAND-RM-704). RAND PROJECT AIR FORCE SANTA MONICA CA.
- [34] Cohen, W. W. (1996, March). Learning rules that classify e-mail. In *AAAI spring symposium on machine learning in information access* (Vol. 18, p. 25).
- [35] Helfman, J. I., & Isbell, C. L. (1995). Ishmail: Immediate identification of important information. In *AT&T Labs*
- [36] Rennie, J. (2000, August). ifile: An application of machine learning to e-mail filtering. In *Proc. KDD 2000 Workshop on Text Mining, Boston, MA*.
- [37] Cohen, W. W., & Singer, Y. (1999). Context-sensitive learning methods for text categorization. *ACM Transactions on Information Systems (TOIS)*, 17(2), 141-173.
- [38] Androustopoulos, I., Koutsias, J., Chandrinou, K. V., & Spyropoulos, C. D. (2000, July). An experimental comparison of naive Bayesian and keyword-based anti-spam filtering with personal e-mail messages. In *Proceedings of the 23rd annual*

international ACM SIGIR conference on Research and development in information retrieval (pp. 160-167). ACM.

[39] Hovold, J. (2005, July). Naive Bayes Spam Filtering Using Word-Position-Based Attributes. In *CEAS* (pp. 41-48).

[40] Sahami, M., Dumais, S., Heckerman, D., & Horvitz, E. (1998, July). A Bayesian approach to filtering junk e-mail. In *Learning for Text Categorization: Papers from the 1998 workshop* (Vol. 62, pp. 98-105).

[41] Becchi, M., & Crowley, P. (2007, December). An improved algorithm to accelerate regular expression evaluation. In *Proceedings of the 3rd ACM/IEEE Symposium on Architecture for networking and communications systems* (pp. 145-154). ACM.

[42] Mogull, R., & Securosis, L. L. C. (2007). Understanding and selecting a data loss prevention solution. *Technicalreport, SANS Institute*.

[43] Smith, R. (2007, September). An overview of the Tesseract OCR engine. In *Document Analysis and Recognition, 2007. ICDAR 2007. Ninth International Conference on* (Vol. 2, pp. 629-633). IEEE.

[44] Partridge, C. (1986). Mail routing and the domain system.

[45] TCK Turkish Law. (2004, October 12). Retrieved May 18, 2017, from <http://www.mevzuat.gov.tr/MevzuatMetin/1.5.5237.pdf>

[46] Gürsel, İ. Protection of personal data in international law and the general aspects of the Turkish Data Protection Law.

[47] Turkish Law on the Protection of Personal Data no. 6698. (n.d.). Retrieved May 18, 2017, from <http://www.kisiselverilerinkorunmasi.org/ingilizce-ceviri/>

[48] DataLossDB. (n.d.). Retrieved May 18, 2017, from <https://blog.datalosssdb.org/>

[49] 2016 Reported Data Breaches Expose Over 4 Billion Records. (n.d.). Retrieved May 18, 2017, from <https://www.riskbasedsecurity.com/2017/01/2016-reported-data-breaches-expose-over-4-billion-records/>

[50] Y. (2015, January 14). GitHub Repository, Ybur-yug/python_ocr_tutorial. Retrieved May 18, 2017, from https://github.com/ybur-yug/python_ocr_tutorial

- [51] PdfImages Manual Page. (n.d.). Retrieved May 18, 2017, from http://linuxcommand.org/man_pages/pdfimages1.html
- [52] Walker, G. (n.d.). E-mail Relay Developer Guide. Retrieved May 18, 2017, from <http://emailrelay.sourceforge.net/developer.html>
- [53] Crochemore, M., & Rytter, W. (1994). *Text algorithms*. Maxime Crochemore.
- [54] Mogull, R. (2008). Dlp content discovery: Best practices for stored data discovery and protection. *USA: Securosis, LLC*.
- [55] Stamati-Koromina, V., Ilioudis, C., Overill, R., Georgiadis, C. K., & Stamatis, D. (2012, September). Insider threats in corporate environments: a case study for data leakage prevention. In *Proceedings of the Fifth Balkan Conference in Informatics* (pp. 271-274). ACM.
- [56] Mattmann, C., & Zitting, J. (2011). *Tika in action*. Manning Publications Co..
- [57] Bidgoli, H. (2006). Handbook of information security, Threats, Vulnerabilities, prevention, detection, and management, Vol. 3.
- [58] Global Data Leakage Middle East Report, 2016. (2017, May 17). Retrieved May 30, 2017, from https://infowatch.com/sites/default/files/docs/infowatch_middle_east_data_leakage_report_2016_eng.pdf
- [59] 2016 Data Breach Investigations Report. (2017, May 30). Retrieved May 31, 2017, from http://www.verizonenterprise.com/resources/reports/rp_DBIR_2016_Report_en_xg.pdf
- [60] Yilmaz, K. (2017, June 04). GitHub Repository, kyilmaz80/emailrelay-dlp. Retrieved June 4, 2017, from <https://github.com/kyilmaz80/emailrelay-dlp>

APPENDIX A: CURRICULUM VITALE

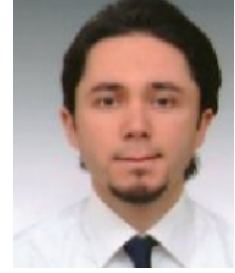
PERSONAL INFORMATION

Surname, Name: YILMAZ, Koray

Date and Place of Birth: 27 Dec 1980,
HILDESHEIM/GERMANY

Marital Status: Single

Email: kyilmaz80@gmail.com



EDUCATION

Degree	Institution	Year of Graduation
M.Sc.	Hacettepe Univ., Physics Engineering	2006
B.Sc.	Hacettepe Univ., Physics Engineering	2003
High School	Private Arı High School	1999

WORK EXPERIENCE

Year	Place	Enrollment
2014- Present	TÜBİTAK	System Admin
2008-2014	TÜBİTAK-BİLGEM/YTE	IT Support Admin
2003-2007	TÜBİTAK-UEKAE/G222	IT Support Admin